

MPLS Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 13, 2014

A. Atlas
K. Tiruveedhula
Juniper Networks
J. Tantsura
Ericsson
IJ. Wijnands
Cisco Systems, Inc.
July 12, 2013

LDP Extensions to Support Maximally Redundant Trees
draft-atlas-mpls-ldp-mrt-00

Abstract

This document specifies extensions to LDP to support the creation of label-switched paths for Maximally Redundant Trees (MRT). A prime use of MRTs is for unicast and multicast IP/LDP Fast-Reroute (MRT-FRR).

The sole protocol extension to LDP is simply the ability to advertise an MRT Capability. This document describes that extension and the associated behavior expected for LSRs and LERS advertising the MRT Capability.

MRT-FRR uses LDP multi-topology extensions and requires three different multi-topology IDs to be allocated from the LDP MT-ID space.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on January 13, 2014.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Requirements Language	3
3. Terminology	3
4. Overview of LDP Signaling Extensions for MRT	4
4.1. MRT Capability Advertisement	5
4.2. Behavior Related to the Rainbow MRT MT-ID	6
4.3. MRT-Blue and MRT-Red FECs	6
5. LDP MRT FEC Advertisements	7
5.1. Downstream Unsolicited Mode	7
5.2. Downstream On Demand Mode	7
5.3. Inter-Area	8
6. Security Considerations	8
7. IANA Considerations	8
8. Acknowledgements	9
9. References	9
9.1. Normative References	9
9.2. Informative References	9
Authors' Addresses	10

1. Introduction

This document describes the LDP signaling extension and associated behavior necessary to support the architecture that defines how IP/LDP Fast-Reroute can use MRTs [I-D.ietf-rtgwg-mrt-frr-architecture]. It is necessary to read the architecture in [I-D.ietf-rtgwg-mrt-frr-architecture] to understand how and why the LDP extensions for behavior are needed.

At least one common standardized algorithm, such as the lowpoint algorithm explained and fully documented in [I-D.enyedi-rtgwg-mrt-frr-algorithm], is required so that the routers supporting MRT computation consistently compute the same MRTs. LDP depends on the IGP to compute the MRTs and alternates; extensions to OSPF are defined in [I-D.atlas-ospf-mrt].

MRT can also be used to protect multicast traffic via either global protection or local protection. [I-D.atlas-rtgwg-mrt-mc-arch] An MRT path can be used to provide node-protection for mLDP traffic via the mechanisms described in [I-D.wijnands-mppls-mldp-node-protection]; an MRT path can also be use to provide link protection for mLDP traffic.

For each destination, IP/LDP Fast-Reroute with MRT (MRT-FRR) creates two alternate destination-based trees separate from the primary next-hop forwarding used during stable operation. LDP uses the multi-topology extensions [I-D.ietf-mppls-ldp-multi-topology] to signal FECs for these two new forwarding topologies, known as MRT-Blue and MRT-Red.

In order to create MRT paths and support IP/LDP Fast-Reroute, a new capability extension is needed for LDP. An LDP implementation supporting MRT must also follow the described rules for originating and managing FECs related to MRT, as indicated by their multi-topology ID. Network reconvergence is described in [I-D.ietf-rtgwg-mrt-frr-architecture] and the worst-case network convergence time can be flooded via the extension in Section 7 of [I-D.atlas-ospf-mrt].

IP/LDP Fast-Reroute using MRTs can provide 100% coverage for link and node failures in an arbitrary network topology where the failure doesn't split the network. It can also be deployed incrementally; an MRT Island is formed of connected supporting routers and the MRTs are computed inside that island.

2. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119]

3. Terminology

For ease of reading, some of the terminology defined in [I-D.ietf-rtgwg-mrt-frr-architecture] is repeated here.

Redundant Trees (RT): A pair of trees where the path from any node X to the root R along the first tree is node-disjoint with the

path from the same node X to the root along the second tree. These can be computed in 2-connected graphs.

Maximally Redundant Trees (MRT): A pair of trees where the path from any node X to the root R along the first tree and the path from the same node X to the root along the second tree share the minimum number of nodes and the minimum number of links. Each such shared node is a cut-vertex. Any shared links are cut-links. Any RT is an MRT but many MRTs are not RTs. The two MRTs are referred to as MRT-Blue and MRT-Red.

MRT Island: From the computing router, the set of routers that support a particular MRT profile and are connected via MRT-eligible links.

MRT-Red: MRT-Red is used to describe one of the two MRTs; it is used to describe the associated forwarding topology and MT-ID. Specifically, MRT-Red is the decreasing MRT where links in the GADAG are taken in the direction from a higher topologically ordered node to a lower one.

MRT-Blue: MRT-Blue is used to describe one of the two MRTs; it is used to describe the associated forwarding topology and MT-ID. Specifically, MRT-Blue is the increasing MRT where links in the GADAG are taken in the direction from a lower topologically ordered node to a higher one.

Rainbow MRT: It is useful to have an MT-ID that refers to the multiple MRT topologies and to the default topology. This is referred to as the Rainbow MRT MT-ID and is used by LDP to reduce signaling and permit the same label to always be advertised to all peers for the same (MT-ID, Prefix).

4. Overview of LDP Signaling Extensions for MRT

Routers need to know which of their neighbors support MRT. Supporting MRT indicates several different aspects of behavior, as listed below.

1. Support for Multi-Topology (MT) - this MAY also be indicated via the Multi-Capability MT Capability [I-D.ietf-mpls-ldp-multi-topology].
2. Understand the Rainbow MRT MT-ID and apply the associated labels to all relevant MT-IDs.
3. Advertise the Rainbow MRT MT-ID to the appropriate neighbors for the associated prefix.

4. If acting as egress for a prefix in the default topology, also advertise and act as egress for the same prefix in MRT-Red and MRT-Blue.
5. For a FEC learned from a neighbor that does not support MRT, originate FECS for MRT-Red and MRT-Blue with the same prefix.

4.1. MRT Capability Advertisement

It is not possible to support MRT without supporting the LDP multi-topology extensions, but it is possible that the only use of the multi-topology extensions is for MRT. In that case, a router MAY not negotiate the multi-topology capability and only negotiate the MRT Capability with its LDP peer. Negotiation of the MT capability is not required with negotiation of the MRT capability.

[EDITOR NOTE: How do we deal with different abilities for IPv4 and IPv6? The MT capability has the Wildcard FEC to indicate this. Do we just assume??]

A new MRT Capability Parameter TLV is defined, which is defined in accordance with LDP Capability definition guidelines[RFC5561].

The LDP MRT capability can be advertised during the LDP session initialization or after the LDP session is established. Advertisement of the MRT capability indicates support of the procedures for establishing the MRT-Blue and MRT-Red LSP paths detailed in this document. If the peer has not advertised the corresponding capability, then it indicates that LSR is not capable of supporting MRT procedures.

The following is the format of the MRT Capability Parameter.

0										1										2										3									
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1								
U F MRT Capability (IANA)										Length (= 1)																													
S Reserved																																							

MRT Capability TLV Format

Where:

U- and F-bits: MUST be 1 and 0, respectively, as per Section 3. (Signaling Extensions) of LDP Capabilities [RFC5561].

MRT Capability: Capability TLV type (IANA assigned)

S-bit: MUST be 1 if used in LDP "Initialization" message. MAY be set to 0 or 1 in dynamic "Capability" message to advertise or withdraw the capability respectively.

Length: The length (in octets) of TLV. Its value is 1.

4.2. Behavior Related to the Rainbow MRT MT-ID

In Section 9 of [I-D.ietf-rtgwg-mrt-frr-architecture], the need to advertise different MPLS labels to different neighbors for the same FEC is described. This can be shortly summarized as either advertising MRT MT-ID differentiated labels to a neighbor or just advertising the same MPLS label for the default topology, for MRT-Red and MRT-Blue. MRT-supporting neighbors in the same domain as the default SPT next-hop get the differentiated MPLS labels; all other neighbors do not.

A second use for the Rainbow MRT MT-ID is for an egress LER to send the Rainbow MRT MT-ID with an IMPLICIT_NULL label to indicate penultimate-hop-popping for all three types of FECs (IP Prefix FEC, MRT-Blue MT-IP Prefix FEC, and MRT-Red MT-IP Prefix FEC).

An LSR advertising the MRT capability MUST recognize the Rainbow MRT MT-ID and associate the advertised label with the specific prefix for the default topology (MT-ID 0) and with the MRT-Red and MRT-Blue MT-IDs associated with all MRT Profiles that advertise LDP as the forwarding mechanism.

An LSR is RECOMMENDED to use the Rainbow MRT MT-ID to reduce the amount of state and signaling required.

As described in [I-D.ietf-rtgwg-mrt-frr-architecture], the recommended experimental value for the Rainbow MRT MT-ID is 3999. The final value will be assigned by IANA and allocated from the LDP MT-ID space.

4.3. MRT-Blue and MRT-Red FECs

To provide MRT support in LDP, the MT Prefix FEC is used. For the default MRT Profile, an MRT-Blue FEC uses the MRT-Blue MT-ID value TBD3 allocated by IANA; for experimental purposes, the value 3998 is suggested. For the default MRT Profile, an MRT-Red FEC uses the MRT-Red MT-ID value TBD2 allocated by IANA; for experimental purposes, the value 3997 is suggested.

The MT Prefix FEC encoding is defined in [I-D.ietf-mpls-ldp-multi-topology] and is used without alternation for signaling MRT-Blue, MRT-Red and Rainbow MRT FECs.

5. LDP MRT FEC Advertisements

This sections describes how and when labels for MRT-Red and MRT-Blue FECs are advertised. The associated LSPs must be created before any failure occurs.

5.1. Downstream Unsolicited Mode

If the upstream session is negotiated with the MRT capability, the Egress LER advertises via a Rainbow MRT FEC an allocated MPLS label; this may be Explicit Null, Implicit Null, or another value.

Based on the MRT algorithm [I-D.enyedi-rtgwg-mrt-frr-algorithm], the IGP computes the MRT-Red and MRT-Blue disjoint paths at Ingress and Transit LSRs. Once the IGP computes the MRT-Red and MRT-Blue next-hops, LDP will advertise the Label Mapping for the MRT-Blue and MRT-Red FECs. If a label is received from a downstream LSR for an MRT-Red or MRT-Blue FEC where the downstream LSR is capable of MRT, the MRT-Red FEC or MRT-Blue FEC label is swapped according to the received downstream label. An LSR may also choose to use the MRT-Red or MRT-Blue path as an alternative for doing fast-reroute for the local traffic.

When a downstream router is not capable of MRT, the LSR is an MRT Island Border Router (IBR) and SHOULD advertise Label Bindings for the MRT-Red FEC and MRT-Blue FEC as well as the associated normal topology. The normal topology's primary next-hops will be used to forward traffic received for the MRT-Red FEC or the MRT-Blue FEC where the FEC's destination is outside the MRT Island. This functionality is critical for partial deployment scenarios.

5.2. Downstream On Demand Mode

After the IGP computes the MRT-Red and MRT-Blue paths, the IGP MAY also decide to use either the MRT-Red or MRT-Blue path as a fast-reroute alternate for the particular FEC. If so, then when in Downstream On Demand Mode, the LSR sends a Label Request for either the MRT-Red or MRT-Blue FEC to the downstream LSR. The downstream LSR responds by either sending a Label Mapping if available or by sending a Label Request to its downstream LSR. Once a Label Mapping is received, the associated label may be used as a fast-reroute alternative to forward IP and LDP traffic.

A Label Mapping may be available in the following circumstances:

- o The LSR is acting as Egress
- o A Label Mapping was already received from its downstream router
- o A Label Mapping for the default topology FEC was received and the downstream router is not capable of MRT or is in a different MRT Island.

5.3. Inter-Area

As discussed in Section 4.2, the Rainbow MRT FEC is defined to facilitate signaling the same label for multiple topologies. Section 9 of [I-D.ietf-rtgwg-mrt-frr-architecture] recommends that traffic leaving an OSPF area or IS-IS level SHOULD use the default topology's shortest-path-tree next-hops instead of remaining on the MRT-Red or MRT-Blue paths. If an LDP peer is in the same OSPF area or IS-IS level as the primary next-hop, then LDP SHOULD advertise different label values for a given set of MRT-Red FEC, MRT-Blue FEC, and FEC, unless Explicit-Null or Implicit-Null is appropriate. If an LDP peer is in a different OSPF area or IS-IS level from the primary next-hop, then LDP SHOULD either advertise the same label value for the given set of MRT-Red FEC, MRT-Blue FEC, and FEC or advertise a single label for the Rainbow MRT FEC, whose behavior is defined in Section 4.2.

6. Security Considerations

This LDP extension is not believed to introduce new security concerns. It relies upon the security architecture already provided for LDP.

7. IANA Considerations

New LDP Capability TLV: "MRT Capability" TLV (requested code point: TBA from LDP registry "TLV Type Name Space"). For interoperable experimental purposes, the value of ... is suggested.

Allocations from the "LDP Multi-Topology (MT) ID Name Space" [I-D.ietf-mpls-ldp-multi-topology] under "LDP Parameter" namespace:

- o Rainbow MRT MT-ID: TBD1
- o default Profile MRT-Red MT-ID: TBD2 - requested under 4096 so it can also be signaled in PIM
- o default Profile MRT-Blue MT-ID: TBD3 - requested under 4096 so it can also be signaled in PIM

For interoperable experiments, the following values are suggested for experimentation: Rainbow MRT MT-ID 3999, default MRT Profile MRT-Blue MT-ID 3998, default MRT Profile MRT-Red MT-ID 3997. The MT-IDs are taken from the 3996-4096 range, which IS-IS defines as for private use, and which [I-D.ietf-mpls-ldp-multi-topology] does not specify as reserved (and MPLS list email suggests that range may be reserved for private use mapping from the IS-IS space).

8. Acknowledgements

The authors would like to thank Ross Callon for his suggestions.

9. References

9.1. Normative References

[I-D.ietf-mpls-ldp-multi-topology]

Zhao, Q., Fang, L., Zhou, C., Li, L., and K. Raza, "LDP Extensions for Multi Topology Routing", draft-ietf-mpls-ldp-multi-topology-08 (work in progress), May 2013.

[I-D.ietf-rtgwg-mrt-frr-architecture]

Atlas, A., Kebler, R., Envedi, G., Csaszar, A., Tantsura, J., Konstantynowicz, M., and R. White, "An Architecture for IP/LDP Fast-Reroute Using Maximally Redundant Trees", draft-ietf-rtgwg-mrt-frr-architecture-03 (work in progress), July 2013.

[RFC5561] Thomas, B., Raza, K., Aggarwal, S., Aggarwal, R., and JL. Le Roux, "LDP Capabilities", RFC 5561, July 2009.

9.2. Informative References

[I-D.atlas-ospf-mrt]

Atlas, A., Hegde, S., Chris, C., and J. Tantsura, "OSPF Extensions to Support Maximally Redundant Trees", draft-atlas-ospf-mrt-00 (work in progress), July 2013.

[I-D.atlas-rtgwg-mrt-mc-arch]

Atlas, A., Kebler, R., Wijnands, I., Csaszar, A., and G. Envedi, "An Architecture for Multicast Protection Using Maximally Redundant Trees", draft-atlas-rtgwg-mrt-mc-arch-02 (work in progress), July 2013.

[I-D.envedi-rtgwg-mrt-frr-algorithm]

Atlas, A., Envedi, G., Csaszar, A., Gopalan, A., and C. Bowers, "Algorithms for computing Maximally Redundant Trees for IP/LDP Fast- Reroute", draft-envedi-rtgwg-mrt-frr-algorithm-03 (work in progress), July 2013.

[I-D.wijnands-mpls-mldp-node-protection]

Wijnands, I., Rosen, E., Raza, K., Tantsura, J., Atlas, A., and Q. Zhao, "mLDP Node Protection", draft-wijnands-mpls-mldp-node-protection-04 (work in progress), June 2013.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC4915] Psena, P., Mirtorabi, S., Roy, A., Nguyen, L., and P. Pillay-Esnault, "Multi-Topology (MT) Routing in OSPF", RFC 4915, June 2007.

[RFC5715] Shand, M. and S. Bryant, "A Framework for Loop-Free Convergence", RFC 5715, January 2010.

Authors' Addresses

Alia Atlas
Juniper Networks
10 Technology Park Drive
Westford, MA 01886
USA

Email: akatlas@juniper.net

Kishore Tiruveedhula
Juniper Networks
10 Technology Park Drive
Westford, MA 01886
USA

Email: kishoret@juniper.net

Jeff Tantsura
Ericsson
300 Holger Way
San Jose, CA 95134
USA

Email: jeff.tantsura@ericsson.com

IJsbrand Wijnands
Cisco Systems, Inc.

Email: ice@cisco.com

OSPF Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 13, 2014

A. Atlas
S. Hegde
C. Bowers
Juniper Networks
J. Tantsura
Ericsson
July 12, 2013

OSPF Extensions to Support Maximally Redundant Trees
draft-atlas-ospf-mrt-00

Abstract

This document specifies extensions to OSPF to support the distributed computation of Maximally Redundant Trees (MRT). Some example uses of the MRTs include IP/LDP Fast-Reroute and global protection or live-live for multicast traffic. The extensions indicate what MRT profile(s) each router supports. Different MRT profiles can be defined to support different uses and to allow transitioning of capabilities. An extension is introduced to flood MRT-Ineligible links, due to administrative policy.

The need for a mechanism to allow routers to advertise a worst-case FIB compute/install time is well understood for controlling convergence. This specification introduces the Controlled Convergence TLV to be carried in the Router Information LSA.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on January 13, 2014.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Requirements Language	3
3. Terminology	3
4. Overview of OSPF Extensions for MRT	4
4.1. Supporting MRT Profiles	4
4.2. GADAG Root Selection	5
4.3. Triggering an MRT Computation	5
5. MRT Capability Advertisement	5
5.1. Advertising MRT Capability in OSPFv2	6
5.2. Advertising MRT Capability in OSPFv3	7
5.3. MRT Profile TLV in Router Information LSA	8
6. Advertising MRT-ineligible links for MRT	8
6.1. MRT-Ineligible Links TLV for OSPFv2	9
6.2. MRT-Ineligible Link TLV for OSPFv3	9
7. Worst-Case Network Convergence Time	10
8. Backwards Compatibility	10
8.1. Handling MRT Capability Changes	11
9. Security Considerations	11
10. IANA Considerations	11
11. References	11
11.1. Normative References	11
11.2. Informative References	12
Authors' Addresses	13

1. Introduction

This document describes the OSPF extensions necessary to support the architecture that defines how IP/LDP Fast-Reroute can use MRTs [I-D.ietf-rtgwg-mrt-frr-architecture]. At least one common standardized algorithm (such as the lowpoint algorithm explained and fully documented in [I-D.enyedi-rtgwg-mrt-frr-algorithm]) is required so that the routers supporting MRT computation consistently compute the same MRTs. MRT can also be used to protect multicast traffic via

either global protection or local protection.[I-D.atlas-rtgwg-mrt-mc-arch]

IP/LDP Fast-Reroute using MRTs can provide 100% coverage for link and node failures in an arbitrary network topology where the failure doesn't split the network. It can also be deployed incrementally inside an OSPF area; an MRT Island is formed of connected supporting routers and the MRTs are computed inside that island.

In the default MRT profile, a supporting router both computes the MRTs and creates the necessary transit forwarding state necessary to provide the two additional forwarding topologies, known as MRT-Blue and MRT-Red.

2. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119]

3. Terminology

For ease of reading, some of the terminology defined in [I-D.ietf-rtgwg-mrt-frr-architecture] is repeated here.

network graph: A graph that reflects the network topology where all links connect exactly two nodes and broadcast links have been transformed into the standard pseudo-node representation.

Redundant Trees (RT): A pair of trees where the path from any node X to the root R along the first tree is node-disjoint with the path from the same node X to the root along the second tree. These can be computed in 2-connected graphs.

Maximally Redundant Trees (MRT): A pair of trees where the path from any node X to the root R along the first tree and the path from the same node X to the root along the second tree share the minimum number of nodes and the minimum number of links. Each such shared node is a cut-vertex. Any shared links are cut-links. Any RT is an MRT but many MRTs are not RTs.

MRT Island: From the computing router, the set of routers that support a particular MRT profile and are connected via MRT-eligible links.

GADAG: Generalized Almost Directed Acyclic Graph - a graph that is the combination of the ADAGs of all blocks. Transforming a network graph into a GADAG is part of the MRT algorithm.

MRT-Red: MRT-Red is used to describe one of the two MRTs; it is used to describe the associated forwarding topology and MT-ID. Specifically, MRT-Red is the decreasing MRT where links in the GADAG are taken in the direction from a higher topologically ordered node to a lower one.

MRT-Blue: MRT-Blue is used to describe one of the two MRTs; it is used to describe the associated forwarding topology and MT-ID. Specifically, MRT-Blue is the increasing MRT where links in the GADAG are taken in the direction from a lower topologically ordered node to a higher one.

4. Overview of OSPF Extensions for MRT

There are two separate aspects that need to be advertised in OSPF. Both derive from the need for all routers supporting an MRT profile to compute the same pair of MRTs to each destination. By executing the same algorithm on the same network graph, distributed routers will compute the same MRTs. Convergence considerations are discussed in [I-D.ietf-rtgwg-mrt-frr-architecture].

The first aspect that must be advertised is which MRT profile(s) are supported and the associated GADAG Root Selection Priority. The second aspect that must be advertised is any links that are not eligible, due to administrative policy, to be part of the MRTs. This must be advertised consistently across the area so that all routers in the MRT Island use the same network graph.

4.1. Supporting MRT Profiles

An MRT Profile defines the exact MRT Algorithm, the MRT-Red MT-ID, the MRT-Blue MT-ID, and the forwarding mechanisms supported for the transit MRT-Red and MRT-Blue forwarding topologies. Finally, the MRT Profile defines exact behavioral rules such as:

- o how reconvergence is handled,
- o inter-area forwarding behavior,

A router that advertises support for an MRT Profile MUST provide the specified forwarding mechanism for its MRT-Red and MRT-Blue forwarding topologies. A router that advertises support for an MRT Profile MUST implement an algorithm that produces the same set of MRT-Red and MRT-Blue next-hops for its MRT-Red and MRT-Blue topologies as is provided by the algorithm specified in the MRT Profile.

A router MAY indicate support for multiple MRT Profiles. A router computes its local MRT Island for each separate MRT Profile that the router supports. The MT-IDs used in one supported MRT Profile MUST NOT overlap with those MT-IDs used in a different supported MRT Profile. Supporting multiple MRT Profiles provides a mechanism for transitioning from one profile to another. Different uses of MRT forwarding topologies may behave better on different MRT profiles.

The default MRT Profile is defined in [I-D.ietf-rtgwg-mrt-frr-architecture]. Its behavior is intended to support IP/LDP unicast and multicast fast-reroute.

4.2. GADAG Root Selection

One aspect of the MRT algorithms is that the selection of the GADAG root can affect the alternates and the traffic through that GADAG root. Therefore, it is important to provide an operator with control over which router will play the role of GADAG root. A measure of the centrality of a node may help determine how good a choice a particular node is.

GADAG Root selection is done using the GADAG Root Selection Priority advertised in the MRT Profile TLV of the Router Information LSA. When the MRTs need to be recalculated, the MRT Island is determined. Inside the set of routers identified as in the MRT Island, routers that are marked as unusable or overloaded (e.g. [RFC3137]) are removed from consideration. Among the remaining routers, the router with the highest GADAG Root Selection Priority is picked to be the GADAG Root. If there are multiple at the same priority, then the router with the highest Router ID is selected.

4.3. Triggering an MRT Computation

When an MRT Computation is triggered, it is triggered for a given MRT Profile in a given area. First, the associated MRT Island is determined. Then, the GADAG Root is selected. Finally, the actual MRT algorithm is run to compute the transit MRT-Red and MRT-Blue topologies. Additionally, the router MAY choose to compute MRT-FRR alternates or make other use of the MRT computation results.

Prefixes can be attached and detached and have their associated MRT-Red and MRT-Blue next-hops computed without requiring a new MRT computation.

5. MRT Capability Advertisement

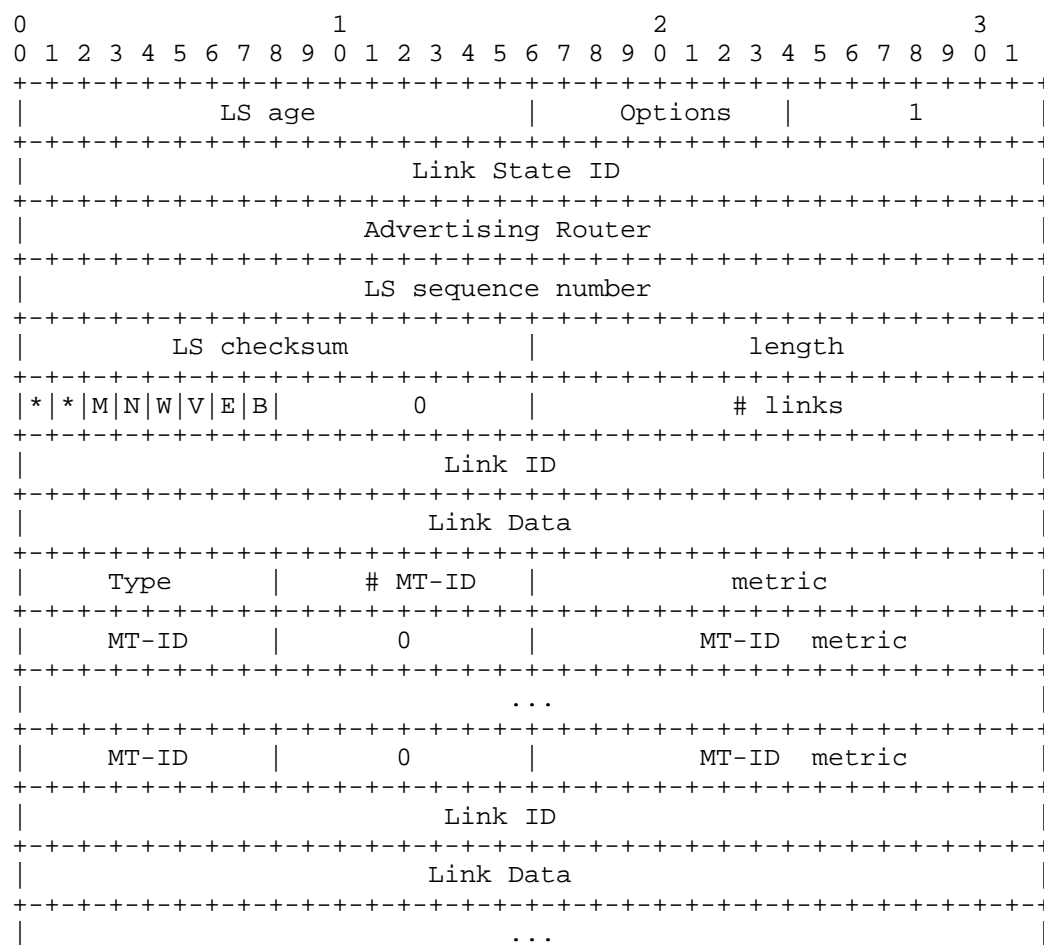
A router that supports MRT indicates this by setting a newly defined M bit in the Router LSA. If the router provides no other information

via a separate MRT Profile TLV, then the router supports the default MRT Profile with a GADAG Root Selection Priority of 100.

In addition, a router can advertise a newly defined MRT Profile TLV within the scope of the OSPF router information LSA [RFC4970]. This TLV also includes the GADAG Root Selection Priority.

5.1. Advertising MRT Capability in OSPFv2

A new M-bit is defined in the Router-LSA (defined in [RFC2328] and updated in [RFC4915]), as pictured below.

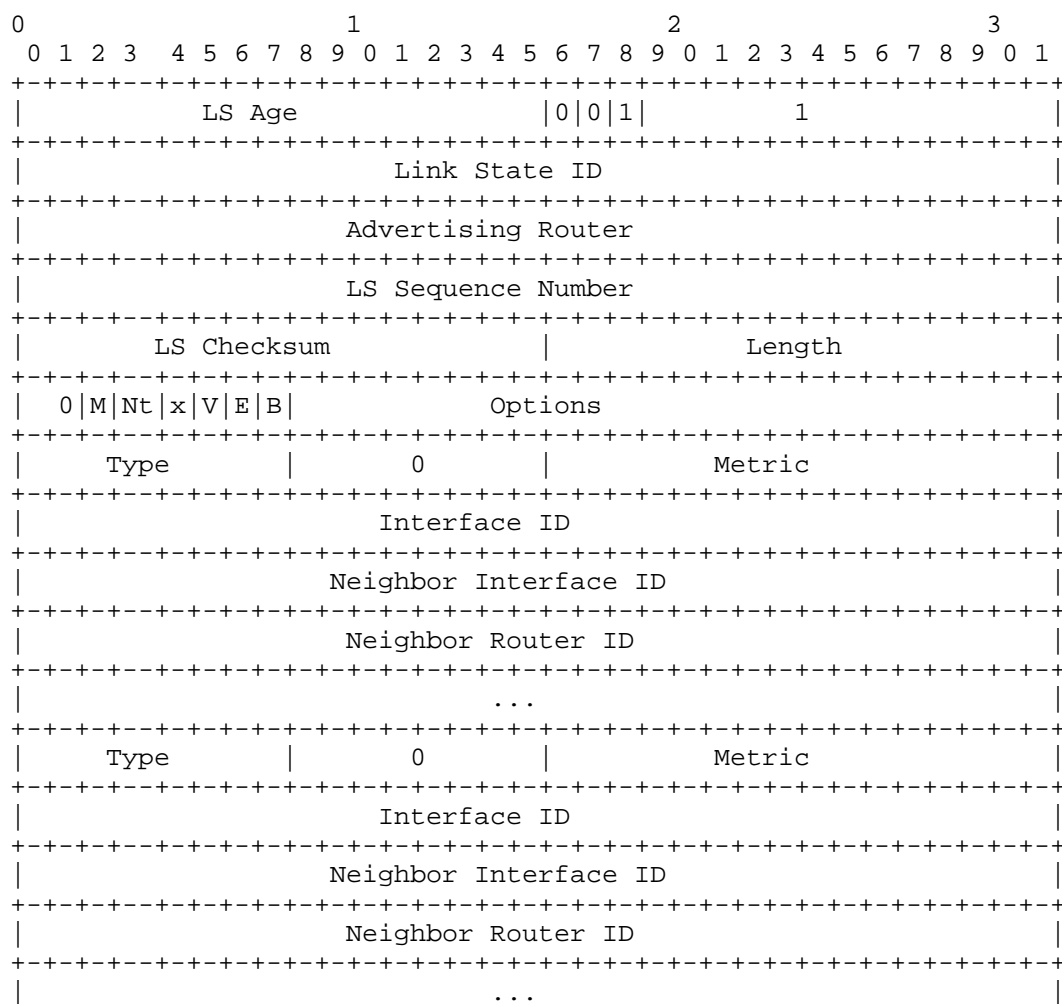


M-bit in OSPFv2 Router LSA

M bit: When set, the router supports MRT. If no separate MRT Profile TLV is advertised in the associated Router Information LSA, then the router supports the default MRT Profile and has a GADAG Root Selection Priority of 100.

5.2. Advertising MRT Capability in OSPFv3

Similarly, the M-bit is defined in the OSPFv3 Router LSA as shown below. Since there can be multiple router LSAs, the M-bit needs to be set on all of them.



M-bit in OSPFv3 Router LSA

M bit: When set, the router supports MRT. If no separate MRT Profile TLV is advertised in the associated Router Information LSA, then the router supports the default MRT Profile and has a GADAG Root Selection Priority of 100.

5.3. MRT Profile TLV in Router Information LSA

A router may advertise an MRT Profile TLV to indicate support for multiple MRT Profiles, for a non-default MRT Profile, and/or to indicate a non-default GADAG Root Selection Priority. The MRT Profile TLV is advertised within the OSPF router information LSA [RFC4970]; the RI LSA MUST have area scope.

TYPE: To Be Allocated by IANA; experimental is 32772

LENGTH: 4 * (number of Profiles)

VALUE:

0										1										2										3									
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1								
Profile ID										GADAG Priority										Reserved																			

Profile ID 0: default MRT Profile

MRT Profile TLV in Router Information LSA

The GADAG Priority is the GADAG Root Selection Priority associated with the advertising router in the MRT Island for the associated MRT Profile, as indicated by the Profile ID. If multiple MRT Profiles are supported, then the length of this TLV varies. The ordering of the profiles inside the TLV is not significant. Multiple appearances of this TLV is not an error.

6. Advertising MRT-ineligible links for MRT

Due to administrative policy, some otherwise eligible links in the network topology may need to be excluded from the network graph upon which the MRT algorithm is run. Since the same network graph must be used across the area, it is critical for OSPF to flood which links to exclude from the MRT calculation. This is done by introducing a new MRT-Ineligible Links TLV to be carried in the Router Information LSA.

If a link is marked by administrative policy as MRT-Ineligible, then a router MUST flood that link in either the MRT-Ineligible TLV or OSPFv3 MRT-Ineligible TLV in the Router Information LSA.

6.1. MRT-Ineligible Links TLV for OSPFv2

MRT-Ineligible links are specified by the Link ID, Link Data, and Type fields, as normally sent in the Router-LSA. See Section A4.1.2 of [RFC2328] for descriptions of these fields.

```

TYPE:  To Be Allocated by IANA; experimental is 32773
LENGTH: 12 * (# of links)
VALUE:
      0                               1                               2                               3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     Link ID                                     |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     Link Data                                     |
+-----+-----+-----+-----+-----+-----+-----+-----+
|      Type      |      Reserved      |
+-----+-----+-----+-----+-----+-----+-----+-----+

```

MRT-Ineligible Links TLV in Router Information LSA

Multiple links can be flooded as MRT-ineligible by listing them inside the same TLV. The ordering of the links in the TLV is not relevant. Multiple appearances of this TLV is not an error.

6.2. MRT-Ineligible Link TLV for OSPFv3

Since links are differently represented in OSPFv2 and OSPFv3, an OSPFv3 MRT-Ineligible Link TLV is defined.

An OSPFv3 MRT-Ineligible Link is defined by its Interface ID, Neighbor Interface ID, Neighbor Router ID, and Type fields. See Appendix A4.1.3 [RFC5340] for the description of these fields.

```

TYPE:  To Be Allocated by IANA; experimental is 32774
LENGTH: 16 * (# of links)
VALUE:
      0                               1                               2                               3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
|      Type      |      Reserved      |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     Interface ID                               |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     Neighbor Interface ID                       |
+-----+-----+-----+-----+-----+-----+-----+-----+

```

```

|                               Neighbor Router ID                               |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+

```

MRT Profile TLV in Router Information LSA

Multiple links can be flooded as MRT-ineligible by listing them inside the same TLV. The ordering of the links in the TLV is not relevant. Multiple appearances of this TLV is not an error.

7. Worst-Case Network Convergence Time

As part of converging the network after a single failure, Section 11.2 of [I-D.ietf-rtgwg-mrt-frr-architecture] describes the need to wait for a configured or advertised period for all routers to be using their new SPTs. Similarly, any work on avoiding micro-forwarding loops during convergence[RFC5715] requires determining the maximum among all routers in the area of the worst-case route computation and FIB installation time. More details on the specific reasoning and need for flooding it are given in [I-D.atlas-bryant-shand-lf-timers].

TYPE: To Be Allocated by IANA; experimental is 32775

LENGTH: 4

VALUE:

```

0                               1                               2                               3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|  Reserved                               |  FIB compute/install time  |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+

```

FIB compute/install time: This is the worst-case time the router may take to compute and install all OSPF routes in the area after a change to a stable network. The value is in milliseconds.

Controlled Convergence TLV in Router Information LSA

The Controlled Convergence TLV is carried in the Router Information LSA and flooded with area-wide scope. A router MUST compute the maximum FIB compute/install time from those flooded in the area. A router MAY use a configured maximum time instead of using and flooding the Controlled Convergence TLV.

8. Backwards Compatibility

The MRT capability bit, the MRT Profile, the MRT-Ineligible Link, and the OSPFv3 MRT-Ineligible Link TLVs are defined in this document. They should not introduce any interoperability issues. Routers that do not support the MRT capability bit in the router LSA SHOULD silently ignore it. Routers that do not support the new MRT-related TLVs SHOULD silently ignore them.

8.1. Handling MRT Capability Changes

When a router changes from supporting MRT to not supporting MRT, it is possible that Router Information LSAs with MRT-related TLVs remain in the neighbors' database briefly. Such MRT-related TLVs SHOULD be ignored when the associated Router LSA from that router does not have the MRT capability set in its Router LSA.

When a router changes from not supporting MRT to supporting MRT, it will flood its Router LSA(s) with the M-bit set and may send an updated Router Information LSA. If a Router LSA is received with the M-bit newly set, an MRT computation SHOULD be scheduled but MAY be delayed up to 60 seconds to allow reception of updated related Router Information LSAs. In general, when changes in MRT-related information is received, an MRT computation SHOULD be triggered.

9. Security Considerations

This OSPF extension is not believed to introduce new security concerns. It relies upon the security architecture already provided for Router LSAs and Router Information LSAs.

10. IANA Considerations

Please allocate a value from the OSPF Router Information TLV Types [RFC4970] for the MRT Profile TLV, for the MRT-Ineligible Link TLV, and for the OSPFv3 MRT-Ineligible Link TLV.

Please create an MRT Profile registry for the MRT Profile TLV. The range is 0 to 255. The default MRT Profile has value 0. Values 1-200 are by Standards Action. Values 201-220 are for experimentation. Values 221-255 are for vendor private use.

11. References

11.1. Normative References

[I-D.enyedi-rtgwg-mrt-frr-algorithm]

Atlas, A., Envedi, G., Csaszar, A., Gopalan, A., and C. Bowers, "Algorithms for computing Maximally Redundant Trees for IP/LDP Fast- Reroute", draft-envedi-rtgwg-mrt-frr-algorithm-03 (work in progress), July 2013.

[I-D.ietf-rtgwg-mrt-frr-architecture]

Atlas, A., Kebler, R., Envedi, G., Csaszar, A., Tantsura, J., Konstantynowicz, M., and R. White, "An Architecture for IP/LDP Fast-Reroute Using Maximally Redundant Trees", draft-ietf-rtgwg-mrt-frr-architecture-03 (work in progress), July 2013.

[RFC2328] Moy, J., "OSPF Version 2", STD 54, RFC 2328, April 1998.

[RFC4970] Lindem, A., Shen, N., Vasseur, JP., Aggarwal, R., and S. Shaffer, "Extensions to OSPF for Advertising Optional Router Capabilities", RFC 4970, July 2007.

[RFC5340] Coltun, R., Ferguson, D., Moy, J., and A. Lindem, "OSPF for IPv6", RFC 5340, July 2008.

11.2. Informative References

[I-D.atlas-bryant-shand-lf-timers]

K, A. and S. Bryant, "Synchronisation of Loop Free Timer Values", draft-atlas-bryant-shand-lf-timers-04 (work in progress), February 2008.

[I-D.atlas-rtgwg-mrt-mc-arch]

Atlas, A., Kebler, R., Wijnands, I., Csaszar, A., and G. Envedi, "An Architecture for Multicast Protection Using Maximally Redundant Trees", draft-atlas-rtgwg-mrt-mc-arch-02 (work in progress), July 2013.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3137] Retana, A., Nguyen, L., White, R., Zinin, A., and D. McPherson, "OSPF Stub Router Advertisement", RFC 3137, June 2001.

[RFC4915] Psenak, P., Mirtorabi, S., Roy, A., Nguyen, L., and P. Pillay-Esnault, "Multi-Topology (MT) Routing in OSPF", RFC 4915, June 2007.

[RFC5715] Shand, M. and S. Bryant, "A Framework for Loop-Free Convergence", RFC 5715, January 2010.

Authors' Addresses

Alia Atlas
Juniper Networks
10 Technology Park Drive
Westford, MA 01886
USA

Email: akatlas@juniper.net

Shraddha Hegde
Juniper Networks

Email: shraddha@juniper.net

Chris Bowers
Juniper Networks

Email: cbowers@juniper.net

Jeff Tantsura
Ericsson
300 Holger Way
San Jose, CA 95134
USA

Email: jeff.tantsura@ericsson.com

Routing Area Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 13, 2014

A. Atlas, Ed.
R. Kebler
Juniper Networks
IJ. Wijnands
Cisco Systems, Inc.
A. Csaszar
G. Enyedi
Ericsson
July 12, 2013

An Architecture for Multicast Protection Using Maximally Redundant Trees
draft-atlas-rtgwg-mrt-mc-arch-02

Abstract

Failure protection is desirable for multicast traffic, whether signaled via PIM or mLDP. Different mechanisms are suitable for different use-cases and deployment scenarios. This document describes the architecture for global protection (aka multicast live-live) and for local protection (aka fast-reroute).

The general methods for global protection and local protection using alternate-trees are dependent upon the use of Maximally Redundant Trees. Local protection can also tunnel traffic in unicast tunnels to take advantage of the routing and fast-reroute mechanisms available for IP/LDP unicast destinations.

The failures protected against are single link or node failures. While the basic architecture might support protection against shared risk group failures, algorithms to dynamically compute MRTs supporting this are for future study.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on January 13, 2014.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	4
1.1. Maximally Redundant Trees (MRTs)	4
1.2. MRTs and Multicast	6
2. Terminology	6
3. Use-Cases and Applicability	8
4. Global Protection: Multicast Live-Live	9
4.1. Creation of MRMTs	10
4.2. Traffic Self-Identification	11
4.2.1. Merging MRMTs for PIM if Traffic Doesn't Self-Identify	12
4.3. Convergence Behavior	13
4.4. Inter-area/level Behavior	14
4.4.1. Inter-area Node Protection with 2 border routers . . .	15
4.4.2. Inter-area Node Protection with > 2 Border Routers . .	16
4.5. PIM	17
4.5.1. Traffic Handling: RPF Checks	17
4.6. mLDP	17
5. Local Repair: Fast-Reroute	17
5.1. PLR-driven Unicast Tunnels	18
5.1.1. Learning the MPs	19
5.1.2. Using Unicast Tunnels and Indirection	19
5.1.3. MP Alternate Traffic Handling	20
5.1.4. Merge Point Reconvergence	21
5.1.5. PLR termination of alternate traffic	21
5.2. MP-driven Unicast Tunnels	21
5.3. MP-driven Alternate Trees	22
6. Acknowledgements	23
7. IANA Considerations	23
8. Security Considerations	23
9. Appendix A	23
9.1. MP-driven Alternate Trees	23
9.1.1. PIM details for Alternate-Trees	26
9.1.2. mLDP details for Alternate-Trees	26
9.1.3. Traffic Handling by PLR	26
9.2. Methods Compared for PIM	27
9.3. Methods Compared for mLDP	27
10. References	27
10.1. Normative References	27
10.2. Informative References	28
Authors' Addresses	29

1. Introduction

This document describes how the algorithms in [I-D.enyedi-rtgwg-mrt-frr-algorithm], which are used in [I-D.ietf-rtgwg-mrt-frr-architecture] for unicast IP/LDP fast-reroute, can be used to provide protection for multicast traffic. It specifically applies to multicast state signaled by PIM[RFC4601] or mLDP[RFC6388]. There are additional protocols that depend upon these (e.g. VPLS, mVPN, etc.) and consideration of the applicability to such traffic will be in a future version.

In this document, global protection is used to refer to the method of having two maximally disjoint multicast trees where traffic may be sent on both and resolved by the receiver. This is similar to the ability with RSVP-TE LSPs to have a primary and a hot standby, except that it can operate in 1+1 mode. This capability is also referred to as multicast live-live and is a generalized form of that discussed in [I-D.ietf-rtgwg-mofrr]. In this document, local protection refers to the method of having alternate ways of reaching the pre-identified merge points upon detection of a local failure. This capability is also referred to as fast-reroute.

This document describes the general architecture, framework, and trade-offs of the different approaches to solving these general problems. It will recommend how to generally provide global protection and local protection for mLDP and PIM traffic. Where protocol extensions are necessary, they will be defined in separate documents as follows.

- o Global 1+1 Protection Using PIM
- o Global 1+1 Protection Using mLDP
- o Local Protection Using mLDP:
[I-D.wijnands-mpls-mldp-node-protection] This document describes how to provide node-protection and the necessary extensions using targeted LDP session.
- o Local Protection Using PIM

1.1. Maximally Redundant Trees (MRTs)

Maximally Redundant Trees (MRTs) are described in [I-D.enyedi-rtgwg-mrt-frr-algorithm]; here we only give a brief description about the concept. A pair of MRTs is a pair of directed spanning trees (red and blue tree) with a common root, directed so that each node can be reached from the root on both trees. Moreover, these trees are redundant, since they are constructed so that no

single link or single node failure can separate any node from the root on both trees, unless that failed link or node is splitting the network into completely separated components (e.g. the link or node was a cut-edge or cut-vertex).

Although for multicast, the arcs (directed links) are directed away from the root instead of towards the root, the same MRT computations are used and apply. This is similar to how multicast uses unicast routing's next-hops as the upstream-hops. Thus this definition slightly differs from the one presented in [I-D.enyedi-rtgwg-mrt-frr-algorithm], since the arcs are directed away and not towards the root. When we need two paths towards a given destination and not two away from it (e.g. for unicast detours for local repair solutions), we only need to reverse the arcs from how they are used for the unicast routing case; thus constructing MRTs towards or away from the root is the same problem. A pair of MRTs is depicted in Figure 1.

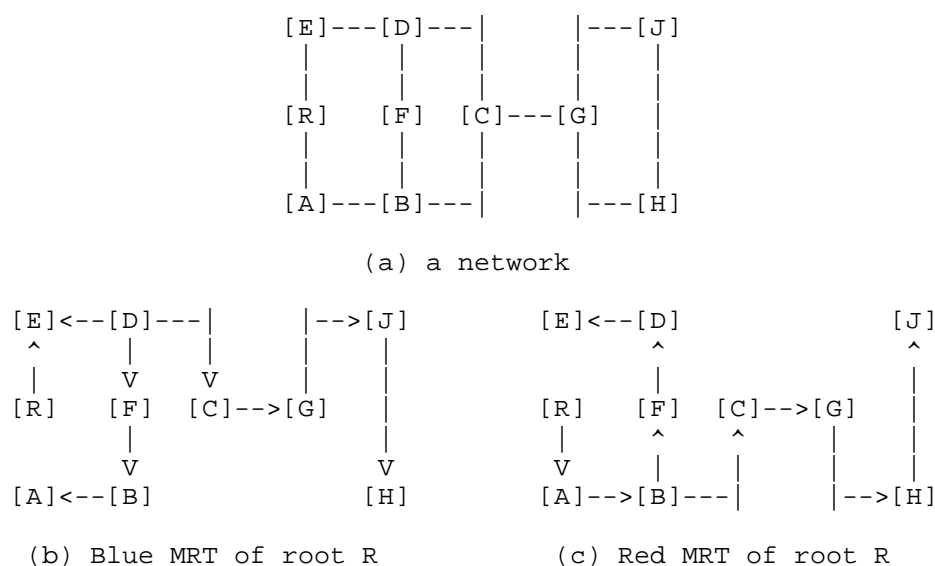


Figure 1: A network and two MRTs found in it

It is important to realize that this redundancy criterion does not imply that, after a failure, either of the MRTs remains intact, since a node failure must affect any spanning tree. Redundancy here means that there will be a set of nodes, which can be reached along the blue MRT, and there will be another set, which remains reachable along the red MRT. As an example, suppose that node F goes down; that would separate B and A on the blue MRT and D and E on the red

MRT. Naturally, it is possible that the intersection of these two sets is not empty, e.g. C, G, H and J will remain reachable on both MRTs. Additionally, observe that a single link can be used in both of the trees in different directions, so even a link failure can cut both trees. In this example, the failure of link $F \leftrightarrow B$ leads to the same reachability sets.

Finally, it is critical to recall that a pair of MRTs is always constructed together and they are not SPTs. While it would be useful to have an algorithm that could find a redundant pair for a given tree (e.g. for the SPT), that is impossible in general. Moreover, if there is a failure and at least one of the trees change, the other tree may need to change as well. Therefore, even if a node still receives the traffic along the red tree, it cannot keep the old red tree, and construct a blue pair for it; there can be reconfiguration in cases when traditional shortest-path-based-thinking would not expect it. To converge to a new pair of disjoint MRTs, it is generally necessary to update both the blue MRT and the red MRT.

The two MRTs provide two separate forwarding topologies that can be used in addition to the default shortest-path-tree (SPT) forwarding topology (usually MT-ID 0). There is a Blue MRT forwarding topology represented by one MT-ID; similarly there is a Red MRT forwarding topology represented by a different MT-ID. Naturally, a multicast protocol is required to use the forwarding topologies information to build the desired multicast trees. The multicast protocol can simply request appropriate upstream interfaces, but include the MT-ID when needed.

1.2. MRTs and Multicast

Maximally Redundant Trees (MRT) provide two advantages for protecting multicast traffic. First, for global protection, MRTs are precisely what needs to be computed to have maximally redundant multicast distribution trees. Second, for local repair, MRTs ensure that there will protection to the merge points; the certainty of a path from any merge point to the PLR that avoids the failure node allows for the creation of alternate trees.

A known disadvantage of MRT, and redundant trees in general, is that the trees do not necessarily provide shortest detour paths. Modeling is underway to investigate and compare the MRT lengths for the different algorithm options [I-D.enyedi-rtgwg-mrt-frr-algorithm].

2. Terminology

2-connected: A graph that has no cut-vertices. This is a graph that requires two nodes to be removed before the network is partitioned.

2-connected cluster: A maximal set of nodes that are 2-connected.

2-edge-connected: A network graph where at least two links must be removed to partition the network.

ADAG: Almost Directed Acyclic Graph - a graph that, if all links incoming to the root were removed, would be a DAG.

block: Either a 2-connected cluster, a cut-edge, or an isolated vertex.

cut-link: A link whose removal partitions the network. A cut-link by definition must be connected between two cut-vertices. If there are multiple parallel links, then they are referred to as cut-links in this document if removing the set of parallel links would partition the network.

cut-vertex: A vertex whose removal partitions the network.

DAG: Directed Acyclic Graph - a graph where all links are directed and there are no cycles in it.

GADAG: Generalized ADAG - a graph that is the combination of the ADAGs of all blocks.

Maximally Redundant Trees (MRT): A pair of trees where the path from any node X to the root R along the first tree and the path from the same node X to the root along the second tree share the minimum number of nodes and the minimum number of links. Each such shared node is a cut-vertex. Any shared links are cut-links. Any RT is an MRT but many MRTs are not RTs.

Maximally Redundant Multicast Trees (MRMT): A pair of multicast trees built of the sub-set of MRTs that is needed to reach all interested receivers.

network graph: A graph that reflects the network topology where all links connect exactly two nodes and broadcast links have been transformed into the standard pseudo-node representation.

Redundant Trees (RT): A pair of trees where the path from any node X to the root R along the first tree is node-disjoint with the path from the same node X to the root along the second tree. These can be computed in 2-connected graphs.

Merge Point (MP): For local repair, a router at which the alternate traffic rejoins the primary multicast tree. For global protection, a router which receives traffic on multiple trees and must decide which stream to forward on.

Point of Local Repair (PLR): The router that detects a local failure and decides whether and when to forward traffic on appropriate alternates.

MT-ID: Multi-topology identifier. The default shortest-path-tree topology is MT-ID 0.

MultiCast Ingress (MCI): Multicast Ingress, the node where the multicast stream enters the current transport technology (MPLS-mLDP or IP-PIM) domain. This maybe the router attached to the multicast source, the PIM Rendezvous Point (RP) or the mLDP Root node address.

Upstream Multicast Hop (UMH): Upstream Multicast Hop, a candidate next-hop that can be used to reach the MCI of the tree.

Stream Selection: The process by which a router determines which of the multiple primary multicast streams to accept and forward. The router can decide on a packet-by-packet basis or simply per-stream. This is done for global protection 1+1 and described in [I-D.ietf-rtgwg-mofrr].

MultiCast Egress (MCE): Multicast Egress, a node where the multicast stream exists the current transport technology (MPLS-mLDP or IP-PIM) domain. This is usually a receiving router that may forward the multicast traffic on towards receivers based upon IGMP or other technology.

3. Use-Cases and Applicability

Protection of multicast streams has gained importance with the use of multicast to distribute video, including live video such as IP-TV. There are a number of different scenarios and uses of multicast that require protection. A few preliminary examples are described below.

- o When video is distributed via IP or MPLS for a cable application, it is desirable to have global protection 1+1 so that the customer-perceived impact is limited. A QAM can join two multicast groups and determine which stream to use based upon the stream quality. A network implementing this may be custom-engineered for this particular purpose.

- o In financial markets, stock ticker data is distributed via multicast. The loss of data can have a significant financial impact. Depending on the network, either global protection 1+1 or local protection can minimize the impact.
- o Several solutions exist for updating software or firmwares of a large number of end-user or operator-owned networking equipment that are based on IP multicast. Since IP multicast is based on datagram transport so taking care of lost data is cumbersome and decreases the advantages offered by multicast. Solutions may rely on sending the updates several times: a properly protected network may result in that less repetitions are required. Other solutions rely on the recipient asking for lost data segments explicitly on-demand. A network failure could cause data loss for a significant number of receivers, which in turn would start requesting the the lost data in a burst that could overload the server. Properly engineered multicast fast reroute would minimise such impacts.
- o Some providers offer multicast VPN services to their customers. SLAs between the customer and provider may set low packet loss requirements. In such cases interruptions longer than the outage timescales targeted by FRR could cause direct financial losses for the provider.

Global protection 1+1 uses maximally redundant multicast trees (MRMTs) to simultaneously distribute a multicast stream on both MRMTs. The disadvantage is the extra state and bandwidth requirements of always sending the traffic twice. The advantage is that the latency of each MRMT can be known and the receiver can select the best stream.

Local protection provides a patch around the fault while the multicast tree reconverges. When PLR replication is used, there is no extra multicast state in the network, but the bandwidth requirements vary based upon how many potential merge-points must be provided. When alternate-trees are used, there is extra multicast state but the bandwidth requirements on a link can be minimized to no more than once for the primary multicast tree traffic and once for the alternate-tree traffic.

4. Global Protection: Multicast Live-Live

In MoFRR [I-D.ietf-rtgwg-mofrr], the idea of joining both a primary and a secondary tree is introduced with the requirement that the primary and secondary trees be link and node disjoint. This works well for networks where there are dual-planes, as explained in [I-D.ietf-rtgwg-mofrr]. For other networks, it is still desirable to

have two disjoint multicast trees and allow a receiver to join both and make its own decision about which traffic to accept.

Using MRTs gives the ability to guarantee that the two trees are as disjoint as possible and dynamically recomputed whenever the topology changes. The MRTs used are rooted at the MultiCast Ingress (MCI). One multicast tree is created using the Blue MRT forwarding topology. The second multicast tree is created using the Red MRT forwarding topology. This can be accomplished by specifying the appropriate MT-ID associated with each forwarding topology.

There are four different aspects of using MRTs for 1+1 Global Protection that are necessary to consider. They are as follows.

1. Creation of the maximally redundant multicast trees (MRMTs) based upon the forwarding topologies.
2. Traffic Identification: How to handle traffic when the two MRMTs overlap due to a cut-vertex or cut-link.
3. Convergence: How to converge after a network change and get back to a protected state.
4. Inter-area/inter-level Behavior: How to compute and use MRMTs when the multicast source is outside the area/level and how to provide border-router protection.

4.1. Creation of MRMTs

The creation of the two maximally redundant multicast trees occurs as described below. This assumes that the next-hops to the MCI associated with the Blue and Red forwarding topologies have already been computed and stored.

1. A receiving router determines that it wants to join both the Blue tree and the Red tree. The details on how it does this decision are not covered in this document and could be based on configuration, additional protocols, etc.
2. The router selects among the Blue next-hops an Upstream Multicast Hop (UMH) to reach the MCI node. The router joins the tree towards the selected UMH including a multi-topology id (MT-ID) identifying the Blue MRT.
3. The router selects among the Red next-hops an Upstream Multicast Hop (UMH) to reach the MCI node. The router joins the tree towards the selected UMH including a multi-topology id (MT-ID) identifying the Red MRT.

4. When a router receives a tree setup request specifying a particular MT-ID (e.g. Color), then the router selects among the Color next-hops to the MCI a UMH node, creates the necessary multicast state, and joins the tree towards the UMH node.

4.2. Traffic Self-Identification

Two maximally redundant trees will share any cut-vertices and cut-links in the network. In the multicast global protection 1+1 case, this means that the potential single failures of the other nodes and links in the network are still protected against. If each cut-vertex cannot associate traffic to a particular MRMT, then the traffic would be incorrectly replicated to both MRMT resulting in complete duplication of traffic. An example of such MRTs is given earlier in Figure 1 and repeated below in Figure 2, where there are two cut-vertices C and G and a cut-link C<->G.

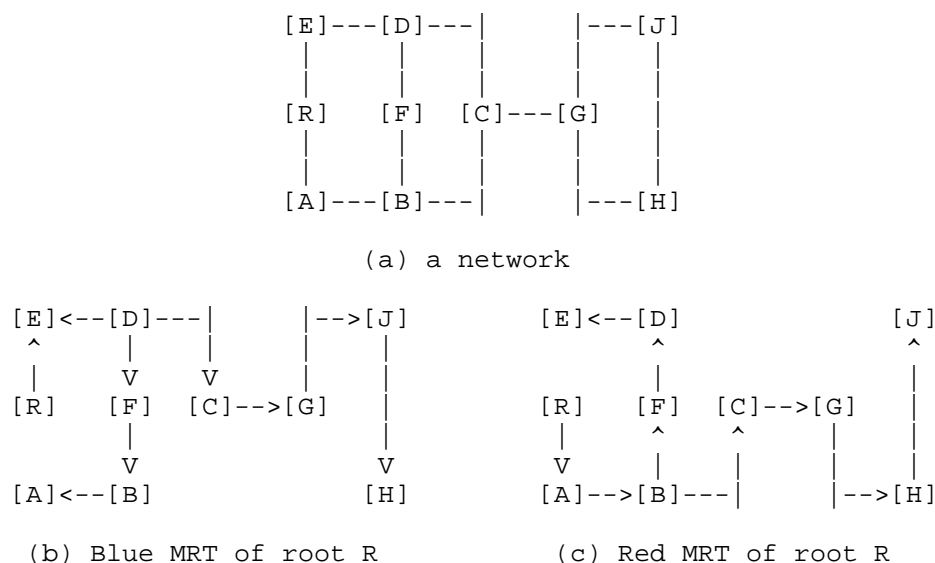


Figure 2: A network and two MRTs found in it

In this example, traffic from the multicast source R to a receiver G, J, or H will cross link C<->G on both the Blue and Red MRMTs. When this occurs, there are several different possibilities depending upon protocol.

mLDP: Different label bindings will be created for the Blue and Red MRMTs. As specified in [I-D.iwijnand-mpls-mldp-multi-topology], the P2MP FEC Element will use the MT IP Address Family to encode the Root node address and MRT T-ID. Each MRMT will therefore have a different P2MP FEC Element and be assigned an independent label.

PIM: There are three different ways to handle IP traffic forwarded based upon PIM when that traffic will overlap on a link.

- A. Different Groups: If different multicast groups are used for each MRMT, then the traffic clearly indicates which MRMT it belongs to. In this case, traffic on the Blue MRMT would use multicast group G-blue and traffic on the Red MRMT would use multicast group G-red.
- B. Different Source Loopbacks: Another option is to use different IP addresses for the source S, so S might announce S-red and S-blue. In this case, traffic on the Blue MRMT would have an IP source of S-blue and traffic on the Red MRMT would have an IP source of S-red.
- C. Stream Selection and Merging: The third option, described in Section 4.2.1, is to have a router that gets (S,G) Joins for both the Blue MT-ID and the Red MT-ID merge those into a single tree. The router may need to select which upstream stream to use, just as if it were a receiving router.

There are three options presented for PIM. The most appropriate will depend upon deployment scenario as well as router capabilities.

4.2.1. Merging MRMTs for PIM if Traffic Doesn't Self-Identify

When traffic doesn't self-identify, the cut-vertices must follow specific rules to avoid traffic duplication. This section describes that behavior which allows the same (S,G) to be used for both the Blue MT-ID and Red MT-ID (e.g. when the traffic doesn't self-identify as to its MT-ID).

The behavior described in this section differs from the conflict resolution described in [RFC6420] because these rules apply to the Global Protection 1+1 case. Specifically, it is not sufficient for a upstream router to pick only one of the two MT-IDs to join because that does not maximize the protection provided.

As described in [RFC6420], a router that receives (S,G) Joins for both the Blue MT-ID and the Red MT-ID can merge the set of downstream interfaces in its forwarding entry. Unlike the procedures defined in [RFC6420], the router must send a Join upstream for each MT-ID. If a

router has different upstream interfaces for these MRMTs, then the router will need to do stream selection and forward the selected stream to its outgoing interfaces, just as if it were an MCE. The stream selection methods of detecting failures and handle traffic discarding are described in [I-D.ietf-rtgwg-mofrr].

This method does not work if the MRMTs merge on a common LAN with different upstream routers. In this case, the traffic cannot be distinguished on the LAN and will result in duplication on the LAN. The normal PIM Assert procedure would stop one of the upstream routers from transmitting duplicates onto the LAN once it is detected. This, in turn, may cause the duplicate stream to be pruned back to the source. Thus, end-to-end protection in this case of the MRMTs converging on a single LAN with different upstream interfaces can only be accomplished by the methods of traffic self-identification.

4.3. Convergence Behavior

It is necessary to handle topology changes and get back to having two MRMTs that provide global protection. To understand the requirements and what can be computed, recall the following facts.

- a. It is not generally possible to compute a single tree that is maximally redundant to an existing tree.
- b. The pair of MRTs must be computed simultaneously.
- c. After a single link or node failure, there is one set of nodes that can be reached from the root on the Blue MRMT and a second set of nodes that can be reached from the root on the Red MRMT. If the failure wasn't a cut-vertex or cut-edge, all nodes will be in at least one of these two sets.

To gracefully converge, it is necessary to never have a router where both its red MRMT and blue MRMT are broken. There are three different ways in which this could be done. These options are being more fully explored to see which is most practical and provides the best set of trade-offs.

Ordered Convergence When a single failure occurs, each receiver determines whether it was affected or unaffected. First, the affected receivers identify the broken MRMT color (e.g. blue) and join the MRMT via their new UMH for that MRT color. Once the affected receivers receive confirmation that the new MRMT has been successfully created back to the MCI, then the affected receivers switch to using that MRMT. The affected receivers tear down the old broken MRMT state and join the MRMT via their new UMH for the

other MRT color (e.g. red). Finally, once the affected receivers receive confirmation that the new MRMT has been successfully created back to the MCI, the affected receivers can tear down the old working MRMT state. Once the affected receivers have updated their state, the unaffected receivers need to also do the same staging - first joining the MRMT via their new UMH for the Blue MRT, waiting for confirmation, switching to using traffic from the Blue MRMT, tearing down the old Blue MRMT state, joining the MRMT via their new UMH for the Red MRT, waiting for confirmation, and tearing down the old Red MRMT state. There are complexities remaining, such as determining how an Unaffected Receiver decides that the Affected Receivers are done. When the topology change isn't a failure, all receivers are unaffected and the same process can apply.

Protocol Make-Before-Break In the control plane, a router joins the tree on the new Blue topology but does not stop receiving traffic on the old Blue topology. Once traffic is observed from the new Blue UMH, then the router accepts traffic on the new Blue UMH and removes the old Blue UMH. This behavior can happen simultaneously with both Blue and Red forwarding topologies. An advantage is that it works regardless of the type of topology change and existing traffic streams aren't broken. Another advantage is that the complexity is limited and this method is well understood. The disadvantage is that the number of traffic-affecting events depends upon the number of hops to the MCI.

Multicast Source Make-Before-Break On a topology change, routers would create new MRMTs using new MRT forwarding state and leaving the old MRMTs as they are. After the new MRMTs are complete, the multicast source could switch from sending on the old MRMTs to sending on the new MRMTs. After a time, the old MRMTs could be torn down. There are a number of details to still investigate.

4.4. Inter-area/level Behavior

A source outside of the IGP area/level can be treated as a proxy node. When the join request reaches a border router (whether ABR for OSPF or LBR for ISIS), that border router needs to determine whether to use the Blue or Red forwarding topology in the next selected area/level.

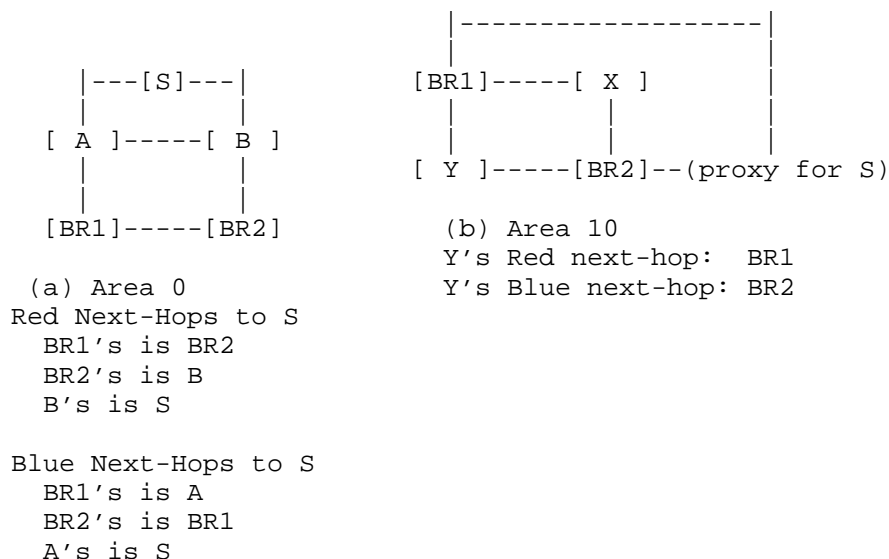


Figure 3: Inter-area Selection - next-hops towards S

Achieving maximally node-disjoint trees across multiple areas is hard due to the information-hiding and abstraction. If there is only one border router, it is trivial but protection of the border router is not possible. With exactly 2 border routers, inter-area/level node protection is reasonably straightforward but can require that the BR rewrite the (S,G) for PIM. With more than 2 border routers, inter-area node protection is possible at the cost of additional bandwidth and border router complexity. These two solutions are described in the following sub-sections.

4.4.1. Inter-area Node Protection with 2 border routers

If there are exactly two border routers between the areas, then the solution and necessary computation is straightforward. In that specific case, each BR knows that only the other BR must specifically be avoided in the second area when a forwarding topology is selected. As described in [I-D.enyedi-rtgwg-mrt-frr-algorithm], it is possible for a node X to determine whether the Red or Blue forwarding topology should be used to reach a node D while avoiding another node Y.

The results of this computation and the resulting changes in MT-ID from Red to Blue or Blue to Red are illustrated in Figure 3. It shows an example where BR1 must modify joins received from Area 10 for the Red MT-ID to use the Blue MT-ID in Area 0. Similarly, BR2 must modify joins received from Area 10 for the Blue MT-ID to use the

Red MT-ID in Area 0.

For mLDP, modifying the MT-ID in the control-plane is all that is needed. For PIM, if the same (S,G) is used for both the Blue MT-ID and the Red MT-ID, then only control-plane changes are needed. However, for PIM, if different group IDs (e.g. G-red and G-blue) or different source loopback addresses (S-red and S-blue) are used, it is necessary to modify the traffic to reflect the MT-ID included in the join message received on that interface. An alternative could be to use an MPLS label that indicates the MT-ID instead of different group IDs or source loopback addresses.

To summarize the necessary logic, when a BR1 receives a join from a neighbor in area N to a destination D in area M on the Color MT-ID, the BR1:

- a. Identifies the BR2 at the other end of the proxy node in area N.
- b. Determines which forwarding topology may avoid BR2 to reach D in area M. Refer to that as Color-2 MT-ID.
- c. Uses Color-2 MT-ID to determine the next-hops to S. When a join is sent upstream, the MT-ID used is that for Color-2.

4.4.2. Inter-area Node Protection with > 2 Border Routers

If there are more than two BRs between areas, then the problem of ensuring inter-area node-disjointness is not solved. Instead, once a request to join the multicast tree has been received by a BR from an area that isn't closest to the multicast source, the BR must join both the Red MT-ID and the Blue MT-ID in the area closest to the multicast source. Regardless of what single link or node failure happens, each BR will receive the multicast stream. Then, the BR can use the stream-selection techniques specified in [I-D.ietf-rtgwg-mofrr] to pick either the Blue or Red stream and forward it to downstream routers in the other area. Each of the BRs for the other area should be attached to a proxy-node representing the other area.

This approach ensures that a BR will receive the multicast stream in the closest area as long as the single link or node failure isn't a single point of failure. Thus, each area or level is independently protected. The BR is required to be able to select among the multicast streams and, if necessary for PIM, translate the traffic to contain the correct (S,G) for forwarding.

4.5. PIM

Capabilities need to be exchanged to determine that a neighbor supports using MRT forwarding topologies with PIM. Additional signaling extensions are not necessary to PIM to support Global Protection. [RFC6420] already defines how to specify an MT-ID as a Join Attribute.

4.5.1. Traffic Handling: RPF Checks

For PIM, RPF checks would still be enabled by the control plane. The control plane can program different forwarding entries on the G-blue incoming interface and on the G-red incoming interface. The other interfaces would still discard both G-blue and G-red traffic.

The receiver would still need to detect failures and handle traffic discarding as is specified in [I-D.ietf-rtgwg-mofrr].

4.6. mLDP

Capabilities need to be exchanged to determine that a neighbor supports using MRT forwarding topologies with mLDP. The basic mechanisms for mLDP to support multi-topology are already described in [I-D.iwijnand-mppls-mlbp-multi-topology]. It may be desirable to extend the capability defined in this draft to indicate that MRT is or is not supported.

5. Local Repair: Fast-Reroute

Local repair for multicast traffic is different from unicast in several important ways.

- o There is more than a single final destination. The full set of receiving routers may not be known by the PLR and may be extremely large. Therefore, it makes sense to repair to the immediate next-hops for link-repair and the next-next-hops for node-repair. These are the potential merge points (MPs).
- o If a failure cannot be positively identified as a node-failure, then it is important to repair to the immediate next-hops since they may have receivers attached.
- o If a failure cannot be positively identified as a link-failure and node protection is desired, then it is important to repair to the next-next-hops since they may not receive traffic from the immediate next-hops.

- o Updating multicast forwarding state may take significantly longer than updating unicast state, since the multicast state is updated tree by tree based on control-plane signaling.
- o For tunnel-based IP/LDP approaches, neither the PLR nor the MP may be able to specify which interface the alternate traffic will arrive at the MP on. The simplest reason is the unicast forwarding includes the use of ECMP and the path selection is based upon internal router behavior for all paths between the PLR and the MP.

For multicast fast-reroute, there are three different mechanisms that can be used. As long as the necessary signaling is available, these methods can be combined in the same network and even for the same PLR and failure point.

PLR-driven Unicast Tunnels: The PLR learns the set of MPs that need protection. On a failure, the PLR replicates the traffic and tunnels it to each MP using the unicast route. If desired, an RSVP-TE tunnel could be used instead of relying upon unicast routing.

MP-driven Unicast Tunnels: Each MP learns the identity of the PLR. Before failure, each MP independently signals to the PLR the desire for protection and other information to use. On a failure, the PLR replicates the traffic and tunnels it to each MP using the unicast route. If desired, an RSVP-TE tunnel could be used instead of relying upon unicast routing.

MP-driven Alternate Trees: Each MP learns the identity of the PLR and the failure point (node and interface) to be protected against. Each MP selects an upstream interface and forwarding topology where the path will avoid the failure point; each MP signals a join towards that upstream interface to create that state.

Each of these options is described in more detail in their respective sections. Then the methods are compared and contrasted for PIM and for mLDP.

5.1. PLR-driven Unicast Tunnels

With PLR-driven unicast tunnels, the PLR learns the set of merge points (MPs) and, on a locally detected failure, uses the existing unicast routing to tunnel the multicast traffic to those merge points. The failure being protected against may be link or node failure. If unicast forwarding can provide an SRLG-protecting alternate, then SRLG-protection is also possible.

There are five aspects to making this work.

1. PLR needs to learn the MPs and their associated MPLS labels to create protection state.
2. Unicast routing has to offer alternates or have dedicated tunnels to reach the MPs. The PLR encapsulates the multicast traffic and directs it to be forwarded via unicast routing.
3. The MP must identify alternate traffic and decide when to accept and forward it or drop it.
4. When the MP reconverges, it must move to its new UMH using make-before-break so that traffic loss is minimized.
5. The PLR must know when to stop sending traffic on the alternates.

5.1.1. Learning the MPs

If link-protection is all that is desired, then the PLR already knows the identities of the MPs. For node-protection, this is not sufficient. In the PLR-driven case, there is no direct communication possible between the PLR and the next-next-hops on the multicast tree. (For mLDP, when targeted LDP sessions are used, this is considered to be MP-driven and is covered in Section 5.2.)

In addition to learning the identities of the MPs, the PLR must also learn the MPLS label, if any, associated with each MP. For mLDP, a different label should be supplied for the alternate traffic; this allows the MP to distinguish between the primary and alternate traffic. For PIM, an MPLS label is used to identify that traffic is the alternate. The unicast tunnel used to send traffic to the MP may have penultimate-hop-popping done; thus without an explicit MPLS label, there is no certainty that a packet could be conclusively identified as primary traffic or as alternate traffic.

A router must tell its UMH the identity of all downstream multicast routers, and their associated alternate labels, on the particular multicast tree. This clearly requires protocol extensions. The extensions for PIM are given in [I-D.kebler-pim-mrt-protection].

5.1.2. Using Unicast Tunnels and Indirection

The PLR must encapsulate the multicast traffic and tunnel it towards each MP. The key point is how that traffic then reaches the MP. There are basically two possibilities. It is possible that a dedicated RSVP-TE tunnel exists and can be used to reach the MP for just this traffic; such an RSVP-TE tunnel would be explicitly routed

to avoid the failure point. The second possibility is that the packet is tunneled via LDP and uses unicast routing. The second case is explored here.

It is necessary to assume that unicast LDP fast-reroute [I-D.ietf-rtgwg-mrt-frr-architecture][RFC5714][RFC5286] is supported by the PLR. Since multicast convergence takes longer than unicast convergence, the PLR may have two different routes to the MP over time. When the failure happens, the PLR will have an alternate, whether LFA or MRT, to reach the MP. Then the unicast routing converges and the PLR will have a new primary route to the MP. Once the routing has converged, it is important that alternate traffic is no longer carried on the MRT forwarding topologies. This rule allows the MRT forwarding topologies to reconverge and be available for the next failure. Therefore, it is also necessary for the tunneled multicast traffic to move from the alternate route to the new primary route when the PLR reconverges. Therefore, the tunneled multicast traffic should use indirection to obtain the unicast routing's current next-hops to the MP. If physical indirection is not feasible, then when the unicast LIB is updated, the associated multicast alternate tunnel state should be as well.

When the PLR detects a local failure, the PLR replicates each multicast packet, swaps or adds the alternate MPLS label needed by the MP, and finally pushes the appropriate label for the MP based upon the outgoing interface selected by the unicast routing.

For PIM, if no alternate labels are supplied by the MPs, then the multicast traffic could be tunneled in IP. This would require unicast IP fast-reroute.

5.1.3. MP Alternate Traffic Handling

A potential Merge Point must determine when and if to accept alternate traffic. There are two critical components to this decision. First, the MP must know the state of all links to its UMH. This allows the MP to determine whether the multicast stream could be received from the UMH. Second, the MP must be able to distinguish between a normal multicast tree packet and an alternate packet.

The logic is similar for PIM and mLDP, but in PIM there is only one RPF-interface or interface of interest to the UMH. In mLDP, all the directly connected interfaces to the UMH are of interest. When the MP detects a local failure, if that interface was the last connected to the UMH and used for the multicast group, then the MP must rapidly switch from accepting the normal multicast tree traffic to accepting the alternate traffic. This rapid change must happen within the same approximately 50 milliseconds that the PLR switching to send traffic

on the alternate takes and for the same reasons. It does no good for the PLR to send alternate traffic if the MP doesn't accept it when it is needed.

The MP can identify alternate traffic based upon the MPLS label. This will be the alternate label that the MP supplied to its UMH for this purpose.

5.1.4. Merge Point Reconvergence

After a failure, the MP will want to join the multicast tree according to the new topology. It is critical that the MP does this in a way that minimizes the traffic disruption. Whenever paths change, there is also the possibility for a traffic-affecting event due to different latencies. However, traffic impact above that should be avoided.

The MP must do make-before-break. Until the MP knows that its new UMH is fully connected to the MCI, the MP should continue to accept its old alternate traffic. The MP could learn that the new UMH is sufficient either via control-plane mechanisms or data-driven. In the latter case, the reception of traffic from the new UMH can trigger the change-over. If the data-driven approach is used, a time-out to force the switch should apply to handle multicast trees that have long quiet periods.

5.1.5. PLR termination of alternate traffic

The PLR sends traffic on the alternates for a configurable time-out. There is no clean way for the next-hop routers and/or next-next-hop routers to indicate that the traffic is no longer needed.

If better control were desired, each MP could tell its UMH what the desired time-out is. The UMH could forward this to the PLR as well. Then the PLR could send alternate traffic to different MPs based upon the MP's individual timer. This would only be an advantage if some of the MPs were expected to have a longer multicast reconvergence time than others - either due to load or router capabilities.

5.2. MP-driven Unicast Tunnels

MP-driven unicast tunnels are only relevant for mLDP where targeted LDP sessions are feasible. For PIM, there is no mechanism to communicate beyond a router's immediate neighbors; these techniques could work for link-protection, but even then there would not be a way of requesting that the PLR should stop sending traffic.

There are three differences for MP-driven unicast tunnels from PLR-

driven unicast tunnels.

1. The MPs learn the identity of the PLR from their UMH. The PLR does not learn the identities of the MPs.
2. The MPs create direct connections to the PLR and communicate their alternate labels.
3. When the MPs have converged, each explicitly tells the PLR to stop sending alternate traffic.

The first means that a router communicates its UMH to all its downstream multicast hops. Then each MP communicates to the PLR(s) (1 for link-protection and 1 for node-protection) and indicates the multicast tree that protection is desired for and the associated alternate label.

When the PLR learns about a new MP, it adds that MP and associated information to the set of MPs to be protected. On a failure, the PLR does the same behavior as for the PLR-driven unicast tunnels.

After the failure, the MP reconverges using make-before-break. Then the MP explicitly communicates to the PLR(s) that alternate traffic is no longer needed for that multicast tree. When the node-protecting PLR hasn't changed for a MP, it may be necessary to withdraw the old alternate label, which tells the PLR to stop transmitting alternate traffic, and then provide a new alternate label.

5.3. MP-driven Alternate Trees

In this document we have defined different solutions to achieve fast convergence for multicast link and node protection based on MRTs. At a high level these solutions can be separated in Local and Global protections. Alternate Trees, which is a Local protection schema, initially looked like an attractive solution for Multicast node protection since it avoids replicating the packet by the PLR to each of the receivers of the protected node and wasting bandwidth. However, this comes at the expense of extra multicast state and complexity. In order to mitigate the extra multicast state its possible to aggregate the Alternate Trees by creating an Alternate Tree per protected node and reuse it for all the multicast trees going through this node. This further complicates the procedures and upstream assigned labels are required to de-aggregate the trees. With aggregation we are also introducing an unwanted side effect. The receiver population of the aggregated trees will very likely not be the same. That means multicast packets will be forwarded on the Alternate Tree to node(s) that may not have a receiver(s) for the

protected tree. The more protected trees are aggregated, the higher the risk of forwarding unwanted multicast packets, this leads again to waisting bandwidth.

Considering the complexity of this solution and the unwanted side-effects, the authors of this document believe its better to solve Multicast node protection using a Global protection schema, as documented in Section 4. The solution previously defined in this section has been move to Appendix A (Section 9).

6. Acknowledgements

The authors would like to thank Kishore Tiruveedhula, Santosh Esale, and Maciek Konstantynowicz for their suggestions and review.

7. IANA Considerations

This doument includes no request to IANA.

8. Security Considerations

This architecture is not currently believed to introduce new security concerns.

9. Appendix A

9.1. MP-driven Alternate Trees

For some networks, it is highly desirable not to have the PLR perform replication to each MP. PLR replication can cause substantial congestion on links used by alternates to different MPs. At the same time, it is also desirable to have minimal extra state created in the network. This can be resolved by creating alternate-trees that can protect multiple multicast groups as a bypass-alternate-tree. An alternate-tree can also be created per multicast group, PLR and failure point.

It is not possible to merge alternate-trees for different PLRs or for different neighbors. This is shown in Figure 4 where G can't select an acceptable upstream node on the alternate tree that doesn't violate either the need to avoid C (for PLR A) or D (for PLR B).

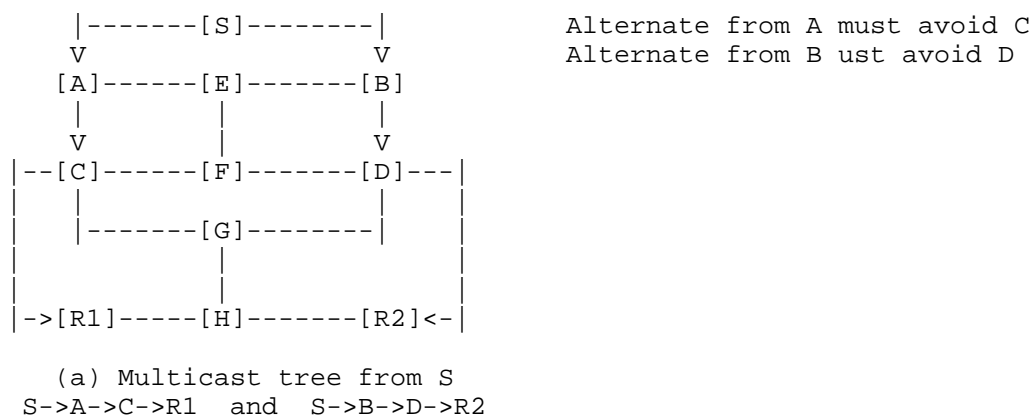


Figure 4: Alternate Trees from PLR A and B can't be merged

A MP that joins an alternate-tree for a particular multicast stream should not expect or request PLR-replicated tunneled alternate traffic for that same multicast stream.

Each alternate-tree is identified by the PLR which sources the traffic and the failure point (node and link) (FP) to be avoided. Different multicast groups with the same PLR and FP may have different sets of MPs - but they are all at most going to include the FP (for link protection) and the neighbors of FP except for the PLR. For a bypass-alternate-tree to work, it must be acceptable to temporarily send a multicast group's traffic to FP's neighbors that do not need it. This is the trade-off required to reduce alternate-tree state and use bypass-alternate-trees. As discussed in Section 5.1.3, a potential MP can determine whether to accept alternate traffic based upon the state of its normal upstream links. Alternate traffic for a group the MP hasn't joined can just be discarded.

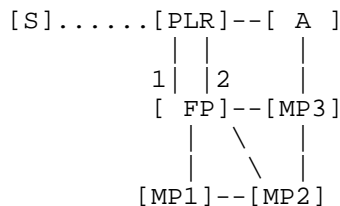


Figure 5: Alternate Tree Scenario

For any router, knowing the PLR and the FP to avoid will force selection of either the Blue MRT or the Red MRT. It is possible that the FP doesn't actually appear in either MRT path, but the FP will always be in either the set of nodes that might be used for the Blue MRT path or the set of nodes that might be used for the Red MRT path. The FP's membership in one of the sets is a function of the partial ordering and topological ordering created by the MRT algorithm and is consistent between routers in the network graph.

To create an alternate-tree, the following must happen:

1. For node-protection, the MP learns from its upstream (the FP) the node-id of its upstream (the PLR) and, optionally, a link identifier for the link used to the PLR. The link-id is only needed for traffic handling in PIM, since mLDp can have targeted sessions between the MP and the PLR.
2. For link-protection, the MP needs to know the node-id of its upstream (the PLR) and, optionally, its identifier for the link used to the PLR.
3. The MP determines whether to use the Blue or Red forwarding topology to reach the PLR while avoiding the FP and associated interface. This gives the MP its alternate-tree upstream interface.
4. The MP signals a backup-join to its alternate-tree upstream interface. The backup-join specifies the PLR, FP and, for PIM, the FP-PLR link identifier. If the alternate-tree is not to be used as a bypass-alternate-tree, then the multicast group (e.g. (S,G) or Opaque-Value) must be specified.
5. A router that receives a backup-join and is not the PLR needs to create multicast state and send a backup-join towards the PLR on the appropriate Blue or Red forwarding topology as is locally determined to avoid the FP and FP-PLR link.
6. Backup-joins for the same (PLR, FP, PLR-FP link-id) that reference the same multicast group can be merged into a single alternate-tree. Similarly, backup-joins for the same (PLR, FP, PLR-FP link-id) that reference no multicast group can be merged into a single alternate-tree.
7. When the PLR receives the backup-join, it associates either the specified multicast group with that alternate-tree, if such is given, or associates all multicast groups that go to the FP via the specified FP-PLR link with the alternate-tree.

For an example, look at Figure 5. FP would send a backup-join to MP3 indicating (PLR, FP, PLR-FP link-1). MP3 sends a backup-join to A. MP1 sends a backup-join to MP2 and MP2 sends a backup-join to MP3.

It is necessary that traffic on each alternate-tree self-identify as to which alternate-tree it is part of. This is because an alternate-tree for a multicast-group and a particular (PLR, FP, PLR-FP link-id) can easily overlap with an alternate-tree for the same multicast group and a different (PLR, FP, PLR-FP link-id). The best way of doing this depends upon whether PIM or mLDP is being used.

9.1.1. PIM details for Alternate-Trees

For PIM, the (S,G) of the IP packet is a globally unique identifier and is understood. To identify the alternate-tree, the most straightforward way is to use MPLS labels distributed in the PIM backup-join messages. A MP can use the incoming label to indicate the set of RPF-interfaces for which the traffic may be an alternate. If the alternate-tree isn't a bypass-alternate-tree, then only one RPF interface is referenced. If the alternate-tree is a bypass-alternate-tree, then multiple RPF-interfaces (parallel links to FP) might be intended. Alternate-tree traffic may cross an interface multiple times - either because the interface is a broadcast interface and different downstream-assigned labels are provided and/or because a MP may provide different labels.

9.1.2. mLDP details for Alternate-Trees

For mLDP, if bypass-alternate-trees are used, then the PLR must provide upstream-assigned labels to each multicast stream. The MP provides the label for the alternate-tree; if the alternate-tree is not a bypass-alternate-tree, this label also describes the multicast stream. If the alternate-tree is a bypass-alternate-tree, then it provides the context for the PLR-assigned labels for each multicast stream. If there are targeted LDP sessions between the PLR and the MPs, then the PLR could provide the necessary upstream-assigned labels.

9.1.3. Traffic Handling by PLR

An issue with traffic is how long should the PLR continue to send alternate traffic out. With an alternate-tree, the PLR can know to stop forwarding alternate traffic on the alternate-tree when that alternate-tree's state is torn down. This provides a clear signal that alternate traffic is no longer needed.

9.2. Methods Compared for PIM

The two approaches that are feasible for PIM are PLR-driven Unicast Tunnels and MP-driven Alternate-Trees.

Aspect	PLR-driven Unicast Tunnels	MP-driven Alternate-Trees
Worst-case Traffic Replication Per Link	1 + number of MPs	2
PLR alternate-traffic	timer-based	control-plane terminated
Extra multicast state	none	per (PLR,FP,S) for bypass mode

Which approach is preferred may be network-dependent. It should also be possible to use both in the same network.

9.3. Methods Compared for mLDP

All three approaches are feasible for mLDP. Below is a brief comparison of various aspects of each.

Aspect	MP-driven Unicast Tunnels	PLR-driven Unicast Tunnels	MP-driven Alternate-Trees
Worst-case Traffic Replication Per Link	1 + number of MPs	1 + number of MPs	2
PLR alternate-traffic	control-plane terminated	timer-based	control-plane terminated
Extra multicast state	none	none	per (PLR,FP,S) for bypass mode

10. References

10.1. Normative References

- [I-D.enyedi-rtgwg-mrt-frr-algorithm]
 Atlas, A., Envedi, G., Csaszar, A., and A. Gopalan,
 "Algorithms for computing Maximally Redundant Trees for

IP/LDP Fast- Reroute",
draft-enyedi-rtgwg-mrt-frr-algorithm-03 (work in
progress), July 2013.

- [I-D.ietf-rtgwg-mrt-frr-architecture]
Atlas, A., Kebler, R., Envedi, G., Csaszar, A., Tantsura, J., Konstantynowicz, M., White, R., and M. Shand, "An Architecture for IP/LDP Fast-Reroute Using Maximally Redundant Trees", draft-ietf-rtgwg-mrt-frr-architecture-03 (work in progress), July 2013.
- [RFC4601] Fenner, B., Handley, M., Holbrook, H., and I. Kouvelas, "Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification (Revised)", RFC 4601, August 2006.
- [RFC6388] Wijnands, IJ., Minei, I., Kompella, K., and B. Thomas, "Label Distribution Protocol Extensions for Point-to-Multipoint and Multipoint-to-Multipoint Label Switched Paths", RFC 6388, November 2011.
- [RFC6420] Cai, Y. and H. Ou, "PIM Multi-Topology ID (MT-ID) Join Attribute", RFC 6420, November 2011.

10.2. Informative References

- [I-D.ietf-rtgwg-mofrr]
Karan, A., Filsfils, C., Farinacci, D., Wijnands, I., Decraene, B., Joerde, U., and W. Henderickx, "Multicast only Fast Re-Route", draft-ietf-rtgwg-mofrr-02 (work in progress), June 2013.
- [I-D.iwijnand-mpls-mldp-multi-topology]
Wijnands, I. and K. Raza, "mLDP Extensions for Multi Topology Routing",
draft-iwijnand-mpls-mldp-multi-topology-03 (work in progress), June 2013.
- [I-D.kebler-pim-mrt-protection]
Kebler, R., Atlas, A., Wijnands, IJ., and G. Enyedi, "PIM Extensions for Protection Using Maximally Redundant Trees", draft-kebler-pim-mrt-protection-00 (work in progress), March 2012.
- [I-D.wijnands-mpls-mldp-node-protection]
Wijnands, I., Rosen, E., Raza, K., Tantsura, J., Atlas, A., and Q. Zhao, "mLDP Node Protection",
draft-wijnands-mpls-mldp-node-protection-04 (work in progress), June 2013.

- [RFC5286] Atlas, A. and A. Zinin, "Basic Specification for IP Fast Reroute: Loop-Free Alternates", RFC 5286, September 2008.
- [RFC5714] Shand, M. and S. Bryant, "IP Fast Reroute Framework", RFC 5714, January 2010.

Authors' Addresses

Alia Atlas (editor)
Juniper Networks
10 Technology Park Drive
Westford, MA 01886
USA

Email: akatlas@juniper.net

Robert Kebler
Juniper Networks
10 Technology Park Drive
Westford, MA 01886
USA

Email: rkebler@juniper.net

IJsbrand Wijnands
Cisco Systems, Inc.

Email: ice@cisco.com

Andras Csaszar
Ericsson
Konyves Kalman krt 11
Budapest 1097
Hungary

Email: Andras.Csaszar@ericsson.com

Gabor Sandor Enyedi
Ericsson
Konyves Kalman krt 11.
Budapest 1097
Hungary

Email: Gabor.Sandor.Enyedi@ericsson.com

Routing Area Working Group
Internet-Draft
Intended status: Informational
Expires: April 24, 2014

G. Enyedi, Ed.
A. Csaszar
Ericsson
A. Atlas, Ed.
C. Bowers
Juniper Networks
A. Gopalan
University of Arizona
October 21, 2013

Algorithms for computing Maximally Redundant Trees for IP/LDP Fast-
Reroute
draft-enyedi-rtgwg-mrt-frr-algorithm-04

Abstract

A complete solution for IP and LDP Fast-Reroute using Maximally Redundant Trees is presented in [I-D.ietf-rtgwg-mrt-frr-architecture]. This document defines the associated MRT Lowpoint algorithm that is used in the default MRT profile to compute both the necessary Maximally Redundant Trees with their associated next-hops and the alternates to select for MRT-FRR.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 24, 2014.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents

(<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
2. Terminology and Definitions	4
3. Algorithm Key Concepts	6
3.1. Partial Ordering for Disjoint Paths	6
3.2. Finding an Ear and the Correct Direction	8
3.3. Low-Point Values and Their Uses	11
3.4. Blocks in a Graph	13
3.5. Determining Local-Root and Assigning Block-ID	15
4. Algorithm Sections	16
4.1. MRT Island Identification	17
4.2. Root Selection	18
4.3. Initialization	18
4.4. MRT Lowpoint Algorithm: Computing GADAG using lowpoint inheritance	19
4.5. Augmenting the GADAG by directing all links	21
4.6. Compute MRT next-hops	23
4.6.1. MRT next-hops to all nodes partially ordered with respect to the computing node	24
4.6.2. MRT next-hops to all nodes not partially ordered with respect to the computing node	24
4.6.3. Computing Redundant Tree next-hops in a 2-connected Graph	25
4.6.4. Generalizing for graph that isn't 2-connected	27
4.6.5. Complete Algorithm to Compute MRT Next-Hops	28
4.7. Identify MRT alternates	30
4.8. Finding FRR Next-Hops for Proxy-Nodes	34
5. MRT Lowpoint Algorithm: Complete Specification	36
6. Algorithm Alternatives and Evaluation	37
6.1. Algorithm Evaluation	37
7. Algorithm Work to Be Done	47
8. IANA Considerations	47
9. Security Considerations	47
10. References	47
10.1. Normative References	47
10.2. Informative References	47
Appendix A. Option 2: Computing GADAG using SPF's	49
Appendix B. Option 3: Computing GADAG using a hybrid method	53
Authors' Addresses	55

1. Introduction

MRT Fast-Reroute requires that packets can be forwarded not only on the shortest-path tree, but also on two Maximally Redundant Trees (MRTs), referred to as the MRT-Blue and the MRT-Red. A router which experiences a local failure must also have pre-determined which alternate to use. This document defines how to compute these three things for use in MRT-FRR and describes the algorithm design decisions and rationale. The algorithm is based on those presented in [MRTLinear] and expanded in [EnyediThesis].

Just as packets routed on a hop-by-hop basis require that each router compute a shortest-path tree which is consistent, it is necessary for each router to compute the MRT-Blue next-hops and MRT-Red next-hops in a consistent fashion. This document defines the MRT Lowpoint algorithm to be used as a standard in the default MRT profile for MRT-FRR.

As now, a router's FIB will contain primary next-hops for the current shortest-path tree for forwarding traffic. In addition, a router's FIB will contain primary next-hops for the MRT-Blue for forwarding received traffic on the MRT-Blue and primary next-hops for the MRT-Red for forwarding received traffic on the MRT-Red.

What alternate next-hops a point-of-local-repair (PLR) selects need not be consistent - but loops must be prevented. To reduce congestion, it is possible for multiple alternate next-hops to be selected; in the context of MRT alternates, each of those alternate next-hops would be equal-cost paths.

This document defines an algorithm for selecting an appropriate MRT alternate for consideration. Other alternates, e.g. LFAs that are downstream paths, may be preferred when available and that policy-based alternate selection process[I-D.ietf-rtgwg-lfa-manageability] is not captured in this document.

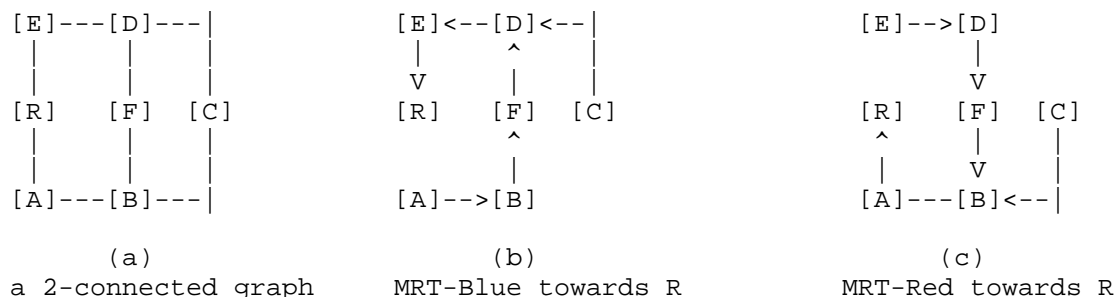
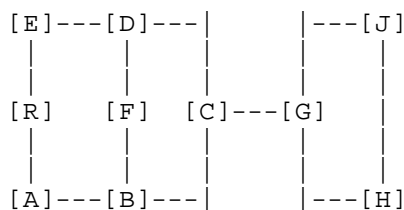
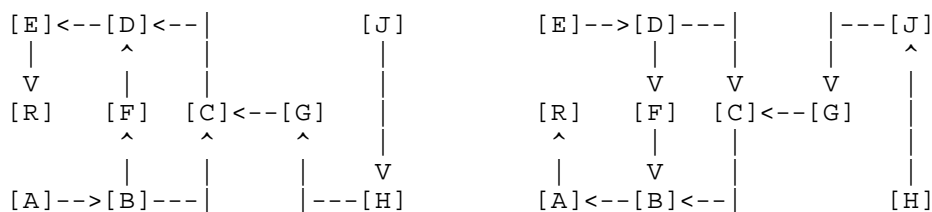


Figure 1

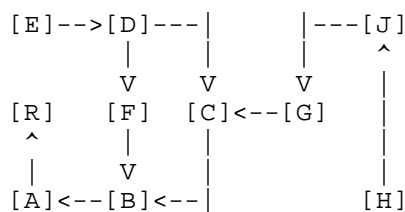
Algorithms for computing MRTs can handle arbitrary network topologies where the whole network graph is not 2-connected, as in Figure 2, as well as the easier case where the network graph is 2-connected (Figure 1). Each MRT is a spanning tree. The pair of MRTs provide two paths from every node X to the root of the MRTs. Those paths share the minimum number of nodes and the minimum number of links. Each such shared node is a cut-vertex. Any shared links are cut-links.



(a) a graph that isn't 2-connected



(b) MRT-Blue towards R



(c) MRT-Red towards R

Figure 2

2. Terminology and Definitions

network graph: A graph that reflects the network topology where all links connect exactly two nodes and broadcast links have been transformed into the standard pseudo-node representation.

Redundant Trees (RT): A pair of trees where the path from any node X to the root R on the first tree is node-disjoint with the path from the same node X to the root along the second tree. These can be computed in 2-connected graphs.

Maximally Redundant Trees (MRT): A pair of trees where the path from any node X to the root R along the first tree and the path from the same node X to the root along the second tree share the minimum number of nodes and the minimum number of links. Each such shared node is a cut-vertex. Any shared links are cut-links. Any RT is an MRT but many MRTs are not RTs.

MRT-Red: MRT-Red is used to describe one of the two MRTs; it is used to describe the associated forwarding topology and MT-ID. Specifically, MRT-Red is the decreasing MRT where links in the GADAG are taken in the direction from a higher topologically ordered node to a lower one.

MRT-Blue: MRT-Blue is used to describe one of the two MRTs; it is used to describe the associated forwarding topology and MT-ID. Specifically, MRT-Blue is the increasing MRT where links in the GADAG are taken in the direction from a lower topologically ordered node to a higher one.

cut-vertex: A vertex whose removal partitions the network.

cut-link: A link whose removal partitions the network. A cut-link by definition must be connected between two cut-vertices. If there are multiple parallel links, then they are referred to as cut-links in this document if removing the set of parallel links would partition the network.

2-connected: A graph that has no cut-vertices. This is a graph that requires two nodes to be removed before the network is partitioned.

spanning tree: A tree containing links that connects all nodes in the network graph.

back-edge: In the context of a spanning tree computed via a depth-first search, a back-edge is a link that connects a descendant of a node *x* with an ancestor of *x*.

2-connected cluster: A maximal set of nodes that are 2-connected. In a network graph with at least one cut-vertex, there will be multiple 2-connected clusters.

block: Either a 2-connected cluster, a cut-edge, or an isolated vertex.

DAG: Directed Acyclic Graph - a digraph containing no directed cycle.

ADAG: Almost Directed Acyclic Graph - a digraph that can be transformed into a DAG with removing a single node (the root node).

GADAG: Generalized ADAG - a digraph, which has only ADAGs as all of its blocks. The root of such a block is the node closest to the global root (e.g. with uniform link costs).

DFS: Depth-First Search

DFS ancestor: A node n is a DFS ancestor of x if n is on the DFS-tree path from the DFS root to x .

DFS descendant: A node n is a DFS descendant of x if x is on the DFS-tree path from the DFS root to n .

ear: A path along not-yet-included-in-the-GADAG nodes that starts at a node that is already-included-in-the-GADAG and that ends at a node that is already-included-in-the-GADAG. The starting and ending nodes may be the same node if it is a cut-vertex.

$X \gg Y$ or $Y \ll X$: Indicates the relationship between X and Y in a partial order, such as found in a GADAG. $X \gg Y$ means that X is higher in the partial order than Y . $Y \ll X$ means that Y is lower in the partial order than X .

$X > Y$ or $Y < X$: Indicates the relationship between X and Y in the total order, such as found via a topological sort. $X > Y$ means that X is higher in the total order than Y . $Y < X$ means that Y is lower in the total order than X .

proxy-node: A node added to the network graph to represent a multi-homed prefix or routers outside the local MRT-fast-reroute-supporting island of routers. The key property of proxy-nodes is that traffic cannot transit them.

3. Algorithm Key Concepts

There are five key concepts that are critical for understanding the MRT Lowpoint algorithm and other algorithms for computing MRTs. The first is the idea of partially ordering the nodes in a network graph with regard to each other and to the GADAG root. The second is the idea of finding an ear of nodes and adding them in the correct direction. The third is the idea of a Low-Point value and how it can be used to identify cut-vertices and to find a second path towards the root. The fourth is the idea that a non-2-connected graph is made up of blocks, where a block is a 2-connected cluster, a cut-edge or an isolated node. The fifth is the idea of a local-root for each node; this is used to compute ADAGs in each block.

3.1. Partial Ordering for Disjoint Paths

Given any two nodes X and Y in a graph, a particular total order means that either $X < Y$ or $X > Y$ in that total order. An example would be a graph where the nodes are ranked based upon their unique IP loopback addresses. In a partial order, there may be some nodes

for which it can't be determined whether $X \ll Y$ or $X \gg Y$. A partial order can be captured in a directed graph, as shown in Figure 3. In a graphical representation, a link directed from X to Y indicates that X is a neighbor of Y in the network graph and $X \ll Y$.

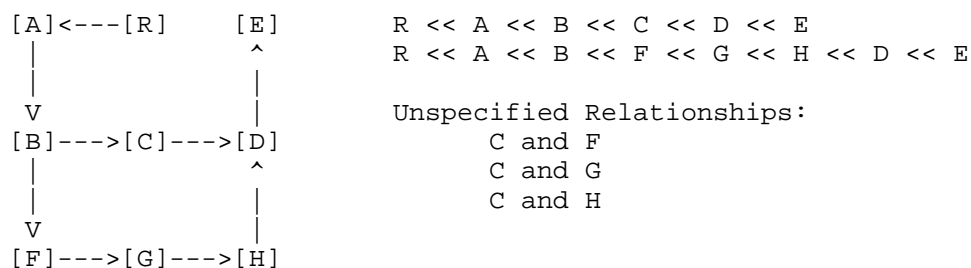


Figure 3: Directed Graph showing a Partial Order

To compute MRTs, the root of the MRTs is at both the very bottom and the very top of the partial ordering. This means that from any node X, one can pick nodes higher in the order until the root is reached. Similarly, from any node X, one can pick nodes lower in the order until the root is reached. For instance, in Figure 4, from G the higher nodes picked can be traced by following the directed links and are H, D, E and R. Similarly, from G the lower nodes picked can be traced by reversing the directed links and are F, B, A, and R. A graph that represents this modified partial order is no longer a DAG; it is termed an Almost DAG (ADAG) because if the links directed to the root were removed, it would be a DAG.

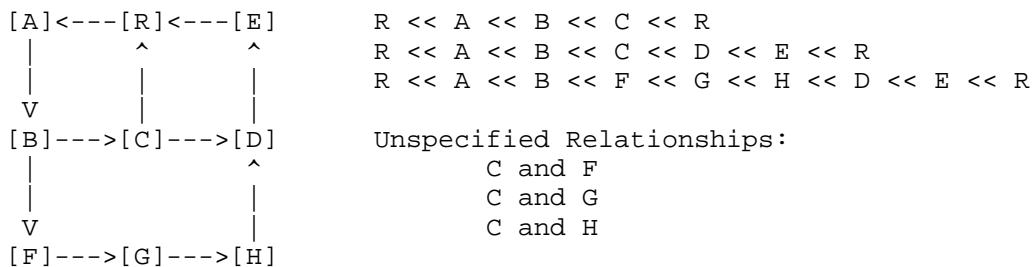


Figure 4: ADAG showing a Partial Order with R lowest and highest

Most importantly, if a node $Y \gg X$, then Y can only appear on the increasing path from X to the root and never on the decreasing path. Similarly, if a node $Z \ll X$, then Z can only appear on the decreasing path from X to the root and never on the increasing path.

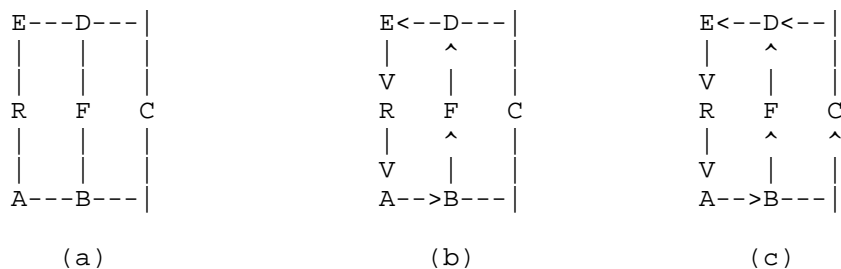
When following the increasing paths, it is possible to pick multiple higher nodes and still have the certainty that those paths will be disjoint from the decreasing paths. E.g. in the previous example node B has multiple possibilities to forward packets along an increasing path: it can either forward packets to C or F .

3.2. Finding an Ear and the Correct Direction

For simplicity, the basic idea of creating a GADAG by adding ears is described assuming that the network graph is a single 2-connected cluster so that an ADAG is sufficient. Generalizing to multiple blocks is done by considering the block-roots instead of the GADAG root - and the actual algorithm is given in Section 4.4.

In order to understand the basic idea of finding an ADAG, first suppose that we have already a partial ADAG, which doesn't contain all the nodes in the block yet, and we want to extend it to cover all the nodes. Suppose that we find a path from a node X to Y such that X and Y are already contained by our partial ADAG, but all the remaining nodes along the path are not added to the ADAG yet. We refer to such a path as an ear.

Recall that our ADAG is closely related to a partial order, more precisely, if we remove root R , the remaining DAG describes a partial order of the nodes. If we suppose that neither X nor Y is the root, we may be able to compare them. If one of them is definitely lesser with respect to our partial order (say $X \ll Y$), we can add the new path to the ADAG in a direction from X to Y . As an example consider Figure 5.



(a) A 2-connected graph
 (b) Partial ADAG (C is not included)
 (c) Resulting ADAG after adding path (or ear) B-C-D

Figure 5

In this partial ADAG, node C is not yet included. However, we can find path B-C-D, where both endpoints are contained by this partial ADAG (we say those nodes are **ready** in the sequel), and the remaining node (node C) is not contained yet. If we remove R, the remaining DAG defines a partial order, and with respect to this partial order we can say that $B \ll D$, so we can add the path to the ADAG in the direction from B to D (arcs B→C and C→D are added). If B were strictly greater than D, we would add the same path in reverse direction.

If in the partial order where an ear's two ends are X and Y, $X \ll Y$, then there must already be a directed path from X to Y already in the ADAG. The ear must be added in a direction such that it doesn't create a cycle; therefore the ear must go from X to Y.

In the case, when X and Y are not ordered with each other, we can select either direction for the ear. We have no restriction since neither of the directions can result in a cycle. In the corner case when one of the endpoints of an ear, say X, is the root (recall that the two endpoints must be different), we could use both directions again for the ear because the root can be considered both as smaller and as greater than Y. However, we strictly pick that direction in which the root is lower than Y. The logic for this decision is explained in Section 4.6

A partial ADAG is started by finding a cycle from the root R back to itself. This can be done by selecting a non-ready neighbor N of R and then finding a path from N to R that doesn't use any links between R and N. The direction of the cycle can be assigned either way since it is starting the ordering.

Once a partial ADAG is already present, we can always add ears to it: just select a non-ready neighbor N of a ready node Q, such that Q is not the root, find a path from N to the root in the graph with Q removed. This path is an ear where the first node of the ear is Q, the next is N, then the path until the first ready node the path reached (that second ready node is the other endpoint of the path). Since the graph is 2-connected, there must be a path from N to R without Q.

It is always possible to select a non-ready neighbor N of a ready node Q so that Q is not the root R . Because the network is 2-connected, N must be connected to two different nodes and only one can be R . Because the initial cycle has already been added to the ADAG, there are ready nodes that are not R . Since the graph is 2-connected, while there are non-ready nodes, there must be a non-ready neighbor N of a ready node that is not R .

```
Generic_Find_Ears_ADAG(root)
  Create an empty ADAG. Add root to the ADAG.
  Mark root as IN_GADAG.
  Select an arbitrary cycle containing root.
  Add the arbitrary cycle to the ADAG.
  Mark cycle's nodes as IN_GADAG.
  Add cycle's non-root nodes to process_list.
  while there exists connected nodes in graph that are not IN_GADAG
    Select a new ear. Let its endpoints be  $X$  and  $Y$ .
    if  $Y$  is root or  $(Y \ll X)$ 
      add the ear towards  $X$  to the ADAG
    else // (a)  $X$  is root or (b)  $X \ll Y$  or (c)  $X, Y$  not ordered
      Add the ear towards  $Y$  to the ADAG
```

Figure 6: Generic Algorithm to find ears and their direction in 2-connected graph

Algorithm Figure 6 merely requires that a cycle or ear be selected without specifying how. Regardless of the way of selecting the path, we will get an ADAG. The method used for finding and selecting the ears is important; shorter ears result in shorter paths along the MRTs. The MRT Lowpoint algorithm's method using Low-Point Inheritance is defined in Section 4.4. Other methods are described in the Appendices (Appendix A and Appendix B).

As an example, consider Figure 5 again. First, we select the shortest cycle containing R , which can be $R-A-B-F-D-E$ (uniform link costs were assumed), so we get to the situation depicted in Figure 5 (b). Finally, we find a node next to a ready node; that must be node C and assume we reached it from ready node B . We search a path from C to R without B in the original graph. The first ready node along this is node D , so the open ear is $B-C-D$. Since $B \ll D$, we add arc $B \rightarrow C$ and $C \rightarrow D$ to the ADAG. Since all the nodes are ready, we stop at this point.

3.3. Low-Point Values and Their Uses

A basic way of computing a spanning tree on a network graph is to run a depth-first-search, such as given in Figure 7. This tree has the important property that if there is a link (x, n) , then either n is a DFS ancestor of x or n is a DFS descendant of x . In other words, either n is on the path from the root to x or x is on the path from the root to n .

```

global_variable: dfs_number

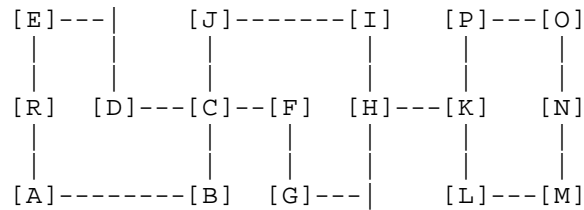
DFS_Visit(node x, node parent)
    D(x) = dfs_number
    dfs_number += 1
    x.dfs_parent = parent
    for each link (x, w)
        if D(w) is not set
            DFS_Visit(w, x)

Run_DFS(node root)
    dfs_number = 0
    DFS_Visit(root, NONE)

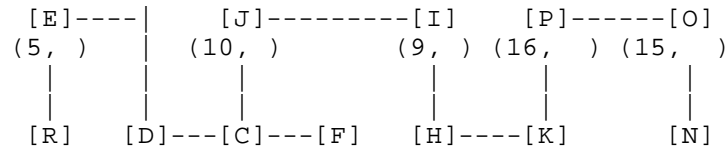
```

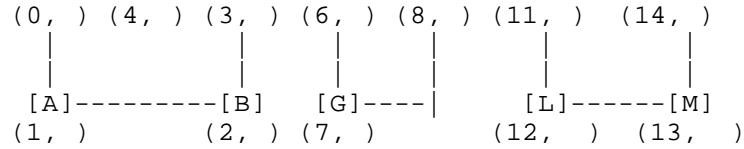
Figure 7: Basic Depth-First Search algorithm

Given a node x , one can compute the minimal DFS number of the neighbours of x , i.e. $\min(D(w) \text{ if } (x,w) \text{ is a link})$. This gives the highest attachment point neighbouring x . What is interesting, though, is what is the highest attachment point from x and x 's descendants. This is what is determined by computing the Low-Point value, as given in Algorithm Figure 9 and illustrated on a graph in Figure 8.

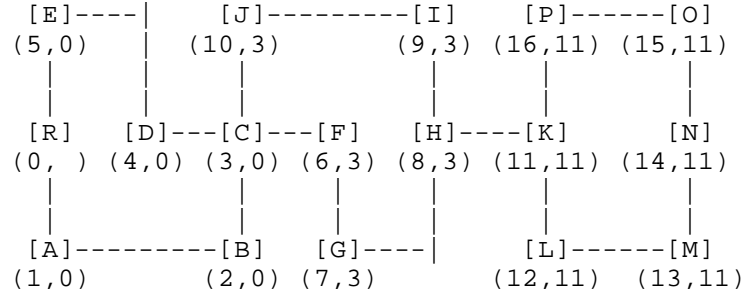


(a) a non-2-connected graph





(b) with DFS values assigned (D(x), L(x))



(c) with low-point values assigned (D(x), L(x))

Figure 8

```

global_variable: dfs_number

Lowpoint_Visit(node x, node parent, interface p_to_x)
    D(x) = dfs_number
    L(x) = D(x)
    dfs_number += 1
    x.dfs_parent = parent
    x.dfs_parent_intf = p_to_x
    x.lowpoint_parent = NONE
    for each interface intf of x:
        if D(intf.remote_node) is not set
            Lowpoint_Visit(intf.remote_node, x, intf)
        if L(intf.remote_node) < L(x)
            L(x) = L(intf.remote_node)
            x.lowpoint_parent = intf.remote_node
            x.lowpoint_parent_intf = intf
        else if intf.remote_node is not parent
            if D(intf.remote_node) < L(x)
                L(x) = D(intf.remote)
                x.lowpoint_parent = intf.remote_node
                x.lowpoint_parent_intf = intf

Run_Lowpoint(node root)
    dfs_number = 0

```

```
Lowpoint_Visit(root, NONE, NONE)
```

Figure 9: Computing Low-Point value

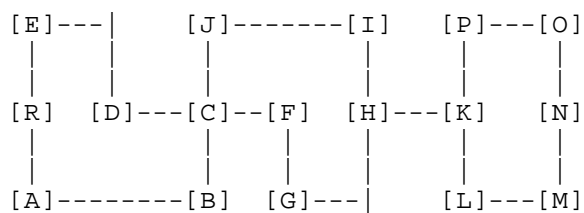
From the low-point value and lowpoint parent, there are two very useful things which motivate our computation.

First, if there is a child c of x such that $L(c) \geq D(x)$, then there are no paths in the network graph that go from c or its descendants to an ancestor of x - and therefore x is a cut-vertex. This is useful because it allows identification of the cut-vertices and thus the blocks. As seen in Figure 8, even if $L(x) < D(x)$, there may be a block that contains both the root and a DFS-child of a node while other DFS-children might be in different blocks. In this example, C 's child D is in the same block as R while F is not.

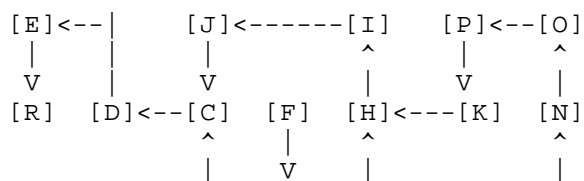
Second, by repeatedly following the path given by `lowpoint_parent`, there is a path from x back to an ancestor of x that does not use the link $[x, x.\text{dfs_parent}]$ in either direction. The full path need not be taken, but this gives a way of finding an initial cycle and then ears.

3.4. Blocks in a Graph

A key idea for an MRT algorithm is that any non-2-connected graph is made up by blocks (e.g. 2-connected clusters, cut-links, and/or isolated nodes). To compute GADAGs and thus MRTs, computation is done in each block to compute ADAGs or Redundant Trees and then those ADAGs or Redundant Trees are combined into a GADAG or MRT.



(a) A graph with four blocks that are:
3 2-connected clusters and a cut-link



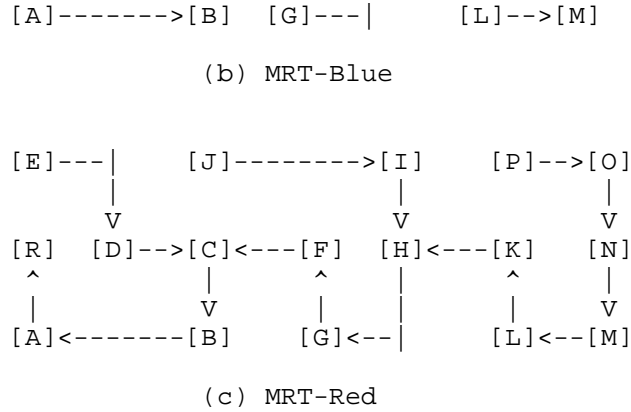


Figure 10

Consider the example depicted in Figure 10 (a). In this figure, a special graph is presented, showing us all the ways 2-connected clusters can be connected. It has four blocks: block 1 contains R, A, B, C, D, E, block 2 contains C, F, G, H, I, J, block 3 contains K, L, M, N, O, P, and block 4 is a cut-edge containing H and K. As can be observed, the first two blocks have one common node (node C) and blocks 2 and 3 do not have any common node, but they are connected through a cut-edge that is block 4. No two blocks can have more than one common node, since two blocks with at least 2 common nodes would qualify as a single 2-connected cluster.

Moreover, observe that if we want to get from one block to another, we must use a cut-vertex (the cut-vertices in this graph are C, H, K), regardless of the path selected, so we can say that all the paths from block 3 along the MRTs rooted at R will cross K first. This observation means that if we want to find a pair of MRTs rooted at R, then we need to build up a pair of RTs in block 3 with K as a root. Similarly, we need to find another one in block 2 with C as a root, and finally, we need the last one in block 1 with R as a root. When all the trees are selected, we can simply combine them; when a block is a cut-edge (as in block 4), that cut-edge is added in the same direction to both of the trees. The resulting trees are depicted in Figure 10 (b) and (c).

Similarly, to create a GADAG it is sufficient to compute ADAGs in each block and connect them.

It is necessary, therefore, to identify the cut-vertices, the blocks and identify the appropriate local-root to use for each block.

3.5. Determining Local-Root and Assigning Block-ID

Each node in a network graph has a local-root, which is the cut-vertex (or root) in the same block that is closest to the root. The local-root is used to determine whether two nodes share a common block.

```

Compute_Localroot(node x, node localroot)
  x.localroot = localroot
  for each DFS child c
    if L(c) < D(x)    //x is not a cut-vertex
      Compute_Localroot(c, x.localroot)
    else
      mark x as cut-vertex
      Compute_Localroot(c, x)

Compute_Localroot(root, root)

```

Figure 11: A method for computing local-roots

There are two different ways of computing the local-root for each node. The stand-alone method is given in Figure 11 and better illustrates the concept; it is used by the MRT algorithms given in the Appendices Appendix A and Appendix B. The method for local-root computation is used in the MRT Lowpoint algorithm for computing a GADAG using Low-Point inheritance and the essence of it is given in Figure 12.

```

Get the current node, s.
Compute an ear(either through lowpoint inheritance
or by following dfs parents) from s to a ready node e.
(Thus, s is not e, if there is such ear.)
if s is e
  for each node x in the ear that is not s
    x.localroot = s
else
  for each node x in the ear that is not s or e
    x.localroot = e.localroot

```

Figure 12: Ear-based method for computing local-roots

Once the local-roots are known, two nodes X and Y are in a common block if and only if one of the following three conditions apply.

- o Y's local-root is X's local-root : They are in the same block and neither is the cut-vertex closest to the root.

- o Y's local-root is X: X is the cut-vertex closest to the root for Y's block
- o Y is X's local-root: Y is the cut-vertex closest to the root for X's block

Once we have computed the local-root for each node in the network graph, we can assign for each node, a block id that represents the block in which the node is present. This computation is shown in Figure 13.

```

global_var: max_block_id

Assign_Block_ID(x, cur_block_id)
  x.block_id = cur_block_id
  foreach DFS child c of x
    if (c.local_root is x)
      max_block_id += 1
      Assign_Block_ID(c, max_block_id)
    else
      Assign_Block_ID(c, cur_block_id)

max_block_id = 0
Assign_Block_ID(root, max_block_id)

```

Figure 13: Assigning block id to identify blocks

4. Algorithm Sections

This algorithm computes one GADAG that is then used by a router to determine its MRT-Blue and MRT-Red next-hops to all destinations. Finally, based upon that information, alternates are selected for each next-hop to each destination. The different parts of this algorithm are described below. These work on a network graph after, for instance, its interfaces are ordered as per Figure 14.

1. Compute the local MRT Island for the particular MRT Profile. [See Section 4.1.]
2. Select the root to use for the GADAG. [See Section 4.2.]
3. Initialize all interfaces to UNDIRECTED. [See Section 4.3.]
4. Compute the DFS value, e.g. $D(x)$, and lowpoint value, $L(x)$. [See Figure 9.]
5. Construct the GADAG. [See Section 4.4]

6. Assign directions to all interfaces that are still `UNDIRECTED`. [See Section 4.5.]
7. From the computing router `x`, compute the next-hops for the MRT-Blue and MRT-Red. [See Section 4.6.]
8. Identify alternates for each next-hop to each destination by determining which one of the blue MRT and the red MRT the computing router `x` should select. [See Section 4.7.]

To ensure consistency in computation, all routers **MUST** order interfaces identically. This is necessary for the DFS, where the selection order of the interfaces to explore results in different trees, and for computing the GADAG, where the selection order of the interfaces to use to form ears can result in different GADAGs. The required ordering between two interfaces from the same router `x` is given in Figure 14.

```
Interface_Compare(interface a, interface b)
  if a.metric < b.metric
    return A_LESS_THAN_B
  if b.metric < a.metric
    return B_LESS_THAN_A
  if a.neighbor.loopback_addr < b.neighbor.loopback_addr
    return A_LESS_THAN_B
  if b.neighbor.loopback_addr < a.neighbor.loopback_addr
    return B_LESS_THAN_A
  // Same metric to same node, so the order doesn't matter anymore.
  // To have a unique, consistent total order,
  // tie-break based on, for example, the link's linkData as
  // distributed in an OSPF Router-LSA
  if a.link_data < b.link_data
    return A_LESS_THAN_B
  return B_LESS_THAN_A
```

Figure 14: Rules for ranking multiple interfaces. Order is from low to high.

4.1. MRT Island Identification

The local MRT Island for a particular MRT profile can be determined by starting from the computing router in the network graph and doing a breadth-first-search (BFS), exploring only links that aren't MRT-ineligible.

```
MRT_Island_Identification(topology, computing_rtr, profile_id)
  for all routers in topology
    rtr.IN_MRT_ISLAND = FALSE
```



```
computing_rtr.IN_MRT_ISLAND = TRUE
explore_list = { computing_rtr }
while (explore_list is not empty)
  next_rtr = remove_head(explore_list)
  for each interface in next_rtr
    if interface is not MRT-ineligible
      if ((interface.remote_node supports profile_id) and
          (interface.remote_node.IN_MRT_ISLAND is FALSE))
        interface.remote_node.IN_MRT_ISLAND = TRUE
        add_to_tail(explore_list, interface.remote_node)
```

Figure 15: MRT Island Identification

4.2. Root Selection

In [I-D.atlas-ospf-mrt], a mechanism is given for routers to advertise the GADAG Root Selection Priority and consistently select a GADAG Root inside the local MRT Island. Before beginning computation, the network graph is reduced to contain only the set of routers that support the specific MRT profile whose MRTs are being computed.

Off-line analysis that considers the centrality of a router may help determine how good a choice a particular router is for the role of GADAG root.

4.3. Initialization

Before running the algorithm, there is the standard type of initialization to be done, such as clearing any computed DFS-values, lowpoint-values, DFS-parents, lowpoint-parents, any MRT-computed next-hops, and flags associated with algorithm.

It is assumed that a regular SPF computation has been run so that the primary next-hops from the computing router to each destination are known. This is required for determining alternates at the last step.

Initially, all interfaces must be initialized to UNDIRECTED. Whether they are OUTGOING, INCOMING or both is determined when the GADAG is constructed and augmented.

It is possible that some links and nodes will be marked as unusable, whether because of configuration, IGP flooding (e.g. MRT-ineligible links in [I-D.atlas-ospf-mrt]), overload, or due to a transient cause such as [RFC3137]. In the algorithm description, it is assumed that such links and nodes will not be explored or used and no more discussion is given of this restriction.

4.4. MRT Lowpoint Algorithm: Computing GADAG using lowpoint inheritance

As discussed in Section 3.2, it is necessary to find ears from a node *x* that is already in the GADAG (known as IN_GADAG). There are two methods to find ears; both are required. The first is by going to a not IN_GADAG DFS-child and then following the chain of low-point parents until an IN_GADAG node is found. The second is by going to a not IN_GADAG neighbor and then following the chain of DFS parents until an IN_GADAG node is found. As an ear is found, the associated interfaces are marked based on the direction taken. The nodes in the ear are marked as IN_GADAG. In the algorithm, first the ears via DFS-children are found and then the ears via DFS-neighbors are found.

By adding both types of ears when an IN_GADAG node is processed, all ears that connect to that node are found. The order in which the IN_GADAG nodes is processed is, of course, key to the algorithm. The order is a stack of ears so the most recent ear is found at the top of the stack. Of course, the stack stores nodes and not ears, so an ordered list of nodes, from the first node in the ear to the last node in the ear, is created as the ear is explored and then that list is pushed onto the stack.

Each ear represents a partial order (see Figure 4) and processing the nodes in order along each ear ensures that all ears connecting to a node are found before a node higher in the partial order has its ears explored. This means that the direction of the links in the ear is always from the node *x* being processed towards the other end of the ear. Additionally, by using a stack of ears, this means that any unprocessed nodes in previous ears can only be ordered higher than nodes in the ears below it on the stack.

In this algorithm that depends upon Low-Point inheritance, it is necessary that every node have a low-point parent that is not itself. If a node is a cut-vertex, that may not yet be the case. Therefore, any nodes without a low-point parent will have their low-point parent set to their DFS parent and their low-point value set to the DFS-value of their parent. This assignment also properly allows an ear between two cut-vertices.

Finally, the algorithm simultaneously computes each node's local-root, as described in Figure 12. This is further elaborated as follows. The local-root can be inherited from the node at the end of the ear unless the end of the ear is *x* itself, in which case the local-root for all the nodes in the ear would be *x*. This is because whenever the first cycle is found in a block, or an ear involving a bridge is computed, the cut-vertex closest to the root would be *x* itself. In all other scenarios, the properties of lowpoint/dfs parents ensure that the end of the ear will be in the same block, and

thus inheriting its local-root would be the correct local-root for all newly added nodes.

The pseudo-code for the GADAG algorithm (assuming that the adjustment of lowpoint for cut-vertices has been made) is shown in Figure 16.

```
Construct_Ear(x, Stack, intf, type)
    ear_list = empty
    cur_node = intf.remote_node
    cur_intf = intf
    not_done = true

    while not_done
        cur_intf.UNDIRECTED = false
        cur_intf.OUTGOING = true
        cur_intf.remote_intf.UNDIRECTED = false
        cur_intf.remote_intf.INCOMING = true

        if cur_node.IN_GADAG is false
            cur_node.IN_GADAG = true
            add_to_list_end(ear_list, cur_node)
            if type is CHILD
                cur_intf = cur_node.lowpoint_parent_intf
                cur_node = cur_node.lowpoint_parent
            else type must be NEIGHBOR
                cur_intf = cur_node.dfs_parent_intf
                cur_node = cur_node.dfs_parent
        else
            not_done = false

    if (type is CHILD) and (cur_node is x)
        //x is a cut-vertex and the local root for
        //the block in which the ear is computed
        localroot = x
    else
        // Inherit local-root from the end of the ear
        localroot = cur_node.localroot
    while ear_list is not empty
        y = remove_end_item_from_list(ear_list)
        y.localroot = localroot
        push(Stack, y)

Construct_GADAG_via_Lowpoint(topology, root)
    root.IN_GADAG = true
    root.localroot = root
    Initialize Stack to empty
    push root onto Stack
    while (Stack is not empty)
```

```

x = pop(Stack)
foreach interface intf of x
    if ((intf.remote_node.IN_GADAG == false) and
        (intf.remote_node.dfs_parent is x))
        Construct_Ear(x, Stack, intf, CHILD)
foreach interface intf of x
    if ((intf.remote_node.IN_GADAG == false) and
        (intf.remote_node.dfs_parent is not x))
        Construct_Ear(x, Stack, intf, NEIGHBOR)

Construct_GADAG_via_Lowpoint(topology, root)

```

Figure 16: Low-point Inheritance GADAG algorithm

4.5. Augmenting the GADAG by directing all links

The GADAG, regardless of the algorithm used to construct it, at this point could be used to find MRTs but the topology does not include all links in the network graph. That has two impacts. First, there might be shorter paths that respect the GADAG partial ordering and so the alternate paths would not be as short as possible. Second, there may be additional paths between a router *x* and the root that are not included in the GADAG. Including those provides potentially more bandwidth to traffic flowing on the alternates and may reduce congestion compared to just using the GADAG as currently constructed.

The goal is thus to assign direction to every remaining link marked as `UNDIRECTED` to improve the paths and number of paths found when the MRTs are computed.

To do this, we need to establish a total order that respects the partial order described by the GADAG. This can be done using Kahn's topological sort [[Kahn_1962_topo_sort](#)] which essentially assigns a number to a node *x* only after all nodes before it (e.g. with a link incoming to *x*) have had their numbers assigned. The only issue with the topological sort is that it works on DAGs and not ADAGs or GADAGs.

To convert a GADAG to a DAG, it is necessary to remove all links that point to a root of block from within that block. That provides the necessary conversion to a DAG and then a topological sort can be done. Finally, all `UNDIRECTED` links are assigned a direction based upon the partial ordering. Any `UNDIRECTED` links that connect to a root of a block from within that block are assigned a direction `INCOMING` to that root. The exact details of this whole process are captured in Figure 17

```

Set_Block_Root_Incoming_Links(topo, root, mark_or_clear)
  foreach node x in topo
    if node x is a cut-vertex or root
      foreach interface i of x
        if (i.remote_node.localroot is x)
          if i.UNDIRECTED
            i.OUTGOING = true
            i.remote_intf.INCOMING = true
            i.UNDIRECTED = false
            i.remote_intf.UNDIRECTED = false
          if i.INCOMING
            if mark_or_clear is mark
              if i.OUTGOING // a cut-edge
                i.STORE_INCOMING = true
                i.INCOMING = false
                i.remote_intf.STORE_OUTGOING = true
                i.remote_intf.OUTGOING = false
                i.TEMP_UNUSABLE = true
                i.remote_intf.TEMP_UNUSABLE = true
              else
                i.TEMP_UNUSABLE = false
                i.remote_intf.TEMP_UNUSABLE = false
            if i.STORE_INCOMING and (mark_or_clear is clear)
              i.INCOMING = true
              i.STORE_INCOMING = false
              i.remote_intf.OUTGOING = true
              i.remote_intf.STORE_OUTGOING = false

Run_Topological_Sort_GADAG(topo, root)
  Set_Block_Root_Incoming_Links(topo, root, MARK)
  foreach node x
    set x.unvisited to the count of x's incoming interfaces
    that aren't marked TEMP_UNUSABLE
  Initialize working_list to empty
  Initialize topo_order_list to empty
  add_to_list_end(working_list, root)
  while working_list is not empty
    y = remove_start_item_from_list(working_list)
    add_to_list_end(topo_order_list, y)
    foreach interface i of y
      if (i.OUTGOING) and (not i.TEMP_UNUSABLE)
        i.remote_node.unvisited -= 1
        if i.remote_node.unvisited is 0
          add_to_list_end(working_list, i.remote_node)
  next_topo_order = 1
  while topo_order_list is not empty
    y = remove_start_item_from_list(topo_order_list)
    y.topo_order = next_topo_order

```

```

        next_topo_order += 1
    Set_Block_Root_Incoming_Links(topo, root, CLEAR)

Add_Undirected_Links(topo, root)
Run_Topological_Sort_GADAG(topo, root)
foreach node x in topo
    foreach interface i of x
        if i.UNDIRECTED
            if x.topo_order < i.remote_node.topo_order
                i.OUTGOING = true
                i.UNDIRECTED = false
                i.remote_intf.INCOMING = true
                i.remote_intf.UNDIRECTED = false
            else
                i.INCOMING = true
                i.UNDIRECTED = false
                i.remote_intf.OUTGOING = true
                i.remote_intf.UNDIRECTED = false

Add_Undirected_Links(topo, root)

```

Figure 17: Assigning direction to UNDIRECTED links

Proxy-nodes do not need to be added to the network graph. They cannot be transited and do not affect the MRTs that are computed. The details of how the MRT-Blue and MRT-Red next-hops are computed and how the appropriate alternate next-hops are selected is given in Section 4.8.

4.6. Compute MRT next-hops

As was discussed in Section 3.1, once a ADAG is found, it is straightforward to find the next-hops from any node X to the ADAG root. However, in this algorithm, we want to reuse the common GADAG and find not only the one pair of MRTs rooted at the GADAG root with it, but find a pair rooted at each node. This is useful since it is significantly faster to compute. It may also provide easier troubleshooting of the MRT-Red and MRT-Blue.

The method for computing differently rooted MRTs from the common GADAG is based on two ideas. First, if two nodes X and Y are ordered with respect to each other in the partial order, then an SPF along OUTGOING links (an increasing-SPF) and an SPF along INCOMING links (a decreasing-SPF) can be used to find the increasing and decreasing paths. Second, if two nodes X and Y aren't ordered with respect to each other in the partial order, then intermediary nodes can be used to create the paths by increasing/decreasing to the intermediary and then decreasing/increasing to reach Y.

As usual, the two basic ideas will be discussed assuming the network is two-connected. The generalization to multiple blocks is discussed in Section 4.6.4. The full algorithm is given in Section 4.6.5.

4.6.1. MRT next-hops to all nodes partially ordered with respect to the computing node

To find two node-disjoint paths from the computing router X to any node Y, depends upon whether $Y \gg X$ or $Y \ll X$. As shown in Figure 18, if $Y \gg X$, then there is an increasing path that goes from X to Y without crossing R; this contains nodes in the interval $[X, Y]$. There is also a decreasing path that decreases towards R and then decreases from R to Y; this contains nodes in the interval $[X, R\text{-small}]$ or $[R\text{-great}, Y]$. The two paths cannot have common nodes other than X and Y.

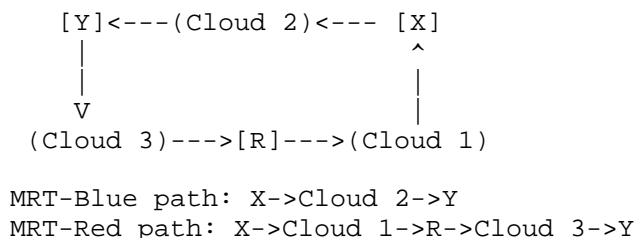


Figure 18: $Y \gg X$

Similar logic applies if $Y \ll X$, as shown in Figure 19. In this case, the increasing path from X increases to R and then increases from R to Y to use nodes in the intervals $[X, R\text{-great}]$ and $[R\text{-small}, Y]$. The decreasing path from X reaches Y without crossing R and uses nodes in the interval $[Y, X]$.

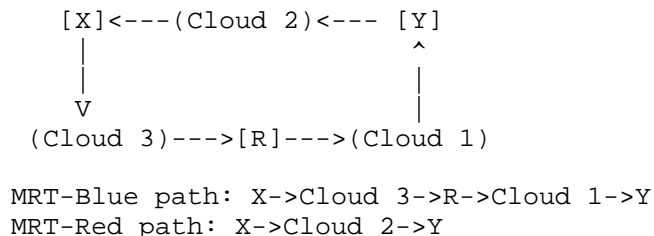
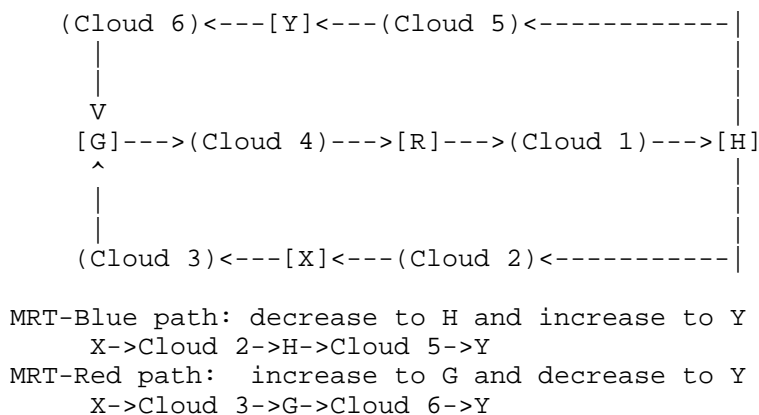


Figure 19: $Y \ll X$

4.6.2. MRT next-hops to all nodes not partially ordered with respect to the computing node

When X and Y are not ordered, the first path should increase until we get to a node G , where $G \gg Y$. At G , we need to decrease to Y . The other path should be just the opposite: we must decrease until we get to a node H , where $H \ll Y$, and then increase. Since R is smaller and greater than Y , such G and H must exist. It is also easy to see that these two paths must be node disjoint: the first path contains nodes in interval $[X, G]$ and $[Y, G]$, while the second path contains nodes in interval $[H, X]$ and $[H, Y]$. This is illustrated in Figure 20. It is necessary to decrease and then increase for the MRT-Blue and increase and then decrease for the MRT-Red; if one simply increased for one and decreased for the other, then both paths would go through the root R .

Figure 20: X and Y unordered

This gives disjoint paths as long as G and H are not the same node. Since $G \gg Y$ and $H \ll Y$, if G and H could be the same node, that would have to be the root R . This is not possible because there is only one incoming interface to the root R which is created when the initial cycle is found. Recall from Figure 6 that whenever an ear was found to have an end that was the root R , the ear was directed from R so that the associated interface on R is outgoing and not incoming. Therefore, there must be exactly one node M which is the largest one before R , so the MRT-Red path will never reach R ; it will turn at M and decrease to Y .

4.6.3. Computing Redundant Tree next-hops in a 2-connected Graph

The basic ideas for computing RT next-hops in a 2-connected graph were given in Section 4.6.1 and Section 4.6.2. Given these two ideas, how can we find the trees?

If some node X only wants to find the next-hops (which is usually the case for IP networks), it is enough to find which nodes are greater and less than X, and which are not ordered; this can be done by running an increasing-SPF and a decreasing-SPF rooted at X and not exploring any links from the ADAG root. (Traversal algorithms other than SPF could safely be used instead where one traversal takes the links in their given directions and the other reverses the links' directions.)

An increasing-SPF rooted at X and not exploring links from the root will find the increasing next-hops to all $Y \gg X$. Those increasing next-hops are X's next-hops on the MRT-Blue to reach Y. An decreasing-SPF rooted at X and not exploring links from the root will find the decreasing next-hops to all $Z \ll X$. Those decreasing next-hops are X's next-hops on the MRT-Red to reach Z. Since the root R is both greater than and less than X, after this increasing-SPF and decreasing-SPF, X's next-hops on the MRT-Blue and on the MRT-Red to reach R are known. For every node $Y \gg X$, X's next-hops on the MRT-Red to reach Y are set to those on the MRT-Red to reach R. For every node $Z \ll X$, X's next-hops on the MRT-Blue to reach Z are set to those on the MRT-Blue to reach R.

For those nodes, which were not reached, we have the next-hops as well. The increasing MRT-Blue next-hop for a node, which is not ordered, is the next-hop along the decreasing MRT-Red towards R and the decreasing MRT-Red next-hop is the next-hop along the increasing MRT-Blue towards R. Naturally, since R is ordered with respect to all the nodes, there will always be an increasing and a decreasing path towards it. This algorithm does not provide the complete specific path taken but just the appropriate next-hops to use. The identity of G and H is not determined.

The final case to considered is when the root R computes its own next-hops. Since the root R is \ll all other nodes, running an increasing-SPF rooted at R will reach all other nodes; the MRT-Blue next-hops are those found with this increasing-SPF. Similarly, since the root R is \gg all other nodes, running a decreasing-SPF rooted at R will reach all other nodes; the MRT-Red next-hops are those found with this decreasing-SPF.



(a) (b)
A 2-connected graph A spanning ADAG rooted at R

Figure 21

As an example consider the situation depicted in Figure 21. There node C runs an increasing-SPF and a decreasing-SPF. The increasing-SPF reaches D, E and R and the decreasing-SPF reaches B, A and R. So towards E the increasing next-hop is D (it was reached through D), and the decreasing next-hop is B (since R was reached through B). Since both D and B, A and R will compute the next hops similarly, the packets will reach E.

We have the next-hops towards F as well: since F is not ordered with respect to C, the MRT-Blue next-hop is the decreasing one towards R (which is B) and the MRT-Red next-hop is the increasing one towards R (which is D). Since B is ordered with F, it will find, for its MRT-Blue, a real increasing next-hop, so packet forwarded to B will get to F on path C-B-F. Similarly, D will have, for its MRT-Red, a real decreasing next-hop, and the packet will use path C-D-F.

4.6.4. Generalizing for graph that isn't 2-connected

If a graph isn't 2-connected, then the basic approach given in Section 4.6.3 needs some extensions to determine the appropriate MRT next-hops to use for destinations outside the computing router X's blocks. In order to find a pair of maximally redundant trees in that graph we need to find a pair of RTs in each of the blocks (the root of these trees will be discussed later), and combine them.

When computing the MRT next-hops from a router X, there are three basic differences:

1. Only nodes in a common block with X should be explored in the increasing-SPF and decreasing-SPF.
2. Instead of using the GADAG root, X's local-root should be used. This has the following implications:
 - a. The links from X's local-root should not be explored.
 - b. If a node is explored in the outgoing SPF so $Y \gg X$, then X's MRT-Red next-hops to reach Y uses X's MRT-Red next-hops to reach X's local-root and if $Z \ll X$, then X's MRT-Blue next-hops to reach Z uses X's MRT-Blue next-hops to reach X's local-root.

- c. If a node W in a common block with X was not reached in the increasing-SPF or decreasing-SPF, then W is unordered with respect to X. X's MRT-Blue next-hops to W are X's decreasing aka MRT-Red next-hops to X's local-root. X's MRT-Red next-hops to W are X's increasing aka Blue MRT next-hops to X's local-root.
- 3. For nodes in different blocks, the next-hops must be inherited via the relevant cut-vertex.

These are all captured in the detailed algorithm given in Section 4.6.5.

4.6.5. Complete Algorithm to Compute MRT Next-Hops

The complete algorithm to compute MRT Next-Hops for a particular router X is given in Figure 22. In addition to computing the MRT-Blue next-hops and MRT-Red next-hops used by X to reach each node Y, the algorithm also stores an "order_proxy", which is the proper cut-vertex to reach Y if it is outside the block, and which is used later in deciding whether the MRT-Blue or the MRT-Red can provide an acceptable alternate for a particular primary next-hop.

```

In_Common_Block(x, y)
    if (((x.localroot is y.localroot) and (x.block_id is y.block_id))
        or (x is y.localroot) or (y is x.localroot))
        return true
    return false

Store_Results(y, direction, spf_root, store_nhs)
    if direction is FORWARD
        y.higher = true
        if store_nhs
            y.blue_next_hops = y.next_hops
    if direction is REVERSE
        y.lower = true
        if store_nhs
            y.red_next_hops = y.next_hops

SPF_No_Traverse_Root(spf_root, block_root, direction, store_nhs)
    Initialize spf_heap to empty
    Initialize nodes' spf_metric to infinity and next_hops to empty
    spf_root.spf_metric = 0
    insert(spf_heap, spf_root)
    while (spf_heap is not empty)
        min_node = remove_lowest(spf_heap)
        Store_Results(min_node, direction, spf_root, store_nhs)
        if ((min_node is spf_root) or (min_node is not block_root))

```

```
    foreach interface intf of min_node
        if (((direction is FORWARD) and intf.OUTGOING) or
            ((direction is REVERSE) and intf.INCOMING) and
            In_Common_Block(spf_root, intf.remote_node))
            path_metric = min_node.spf_metric + intf.metric
            if path_metric < intf.remote_node.spf_metric
                intf.remote_node.spf_metric = path_metric
                if min_node is spf_root
                    intf.remote_node.next_hops = make_list(intf)
                else
                    intf.remote_node.next_hops = min_node.next_hops
                    insert_or_update(spf_heap, intf.remote_node)
            else if path_metric is intf.remote_node.spf_metric
                if min_node is spf_root
                    add_to_list(intf.remote_node.next_hops, intf)
                else
                    add_list_to_list(intf.remote_node.next_hops,
                                     min_node.next_hops)

SetEdge(y)
    if y.blue_next_hops is empty and y.red_next_hops is empty
        if (y.local_root != y) {
            SetEdge(y.localroot)
        }
        y.blue_next_hops = y.localroot.blue_next_hops
        y.red_next_hops = y.localroot.red_next_hops
        y.order_proxy = y.localroot.order_proxy

Compute_MRT_NextHops(x, root)
    foreach node y
        y.higher = y.lower = false
        clear y.red_next_hops and y.blue_next_hops
        y.order_proxy = y
        SPF_No_Traverse_Root(x, x.localroot, FORWARD, TRUE)
        SPF_No_Traverse_Root(x, x.localroot, REVERSE, TRUE)

    // red and blue next-hops are stored to x.localroot as different
    // paths are found via the SPF and reverse-SPF.
    // Similarly any nodes whose local-root is x will have their
    // red_next_hops and blue_next_hops already set.

    // Handle nodes in the same block that aren't the local-root
    foreach node y
        if (y.IN_MRT_ISLAND and (y is not x) and
            (y.localroot is x.localroot) and
            ((y is x.localroot) or (x is y.localroot) or
             (y.block_id is x.block_id)))
            if y.higher
```

```

        y.red_next_hops = x.localroot.red_next_hops
    else if y.lower
        y.blue_next_hops = x.localroot.blue_next_hops
    else
        y.blue_next_hops = x.localroot.red_next_hops
        y.red_next_hops = x.localroot.blue_next_hops

    // Inherit next-hops and order_proxies to other components
    if x is not root
        root.blue_next_hops = x.localroot.blue_next_hops
        root.red_next_hops = x.localroot.red_next_hops
        root.order_proxy = x.localroot
    foreach node y
        if (y is not root) and (y is not x) and y.IN_MRT_ISLAND
            SetEdge(y)

max_block_id = 0
Assign_Block_ID(root, max_block_id)
Compute_MRT_NextHops(x, root)

```

Figure 22

4.7. Identify MRT alternates

At this point, a computing router S knows its MRT-Blue next-hops and MRT-Red next-hops for each destination in the MRT Island. The primary next-hops along the SPT are also known. It remains to determine for each primary next-hop to a destination D, which of the MRTs avoids the primary next-hop node F. This computation depends upon data set in Compute_MRT_NextHops such as each node y's y.blue_next_hops, y.red_next_hops, y.order_proxy, y.higher, y.lower and topo_orders. Recall that any router knows only which are the nodes greater and lesser than itself, but it cannot decide the relation between any two given nodes easily; that is why we need topological ordering.

For each primary next-hop node F to each destination D, S can call Select_Alternates(S, D, F, primary_intf) to determine whether to use the MRT-Blue next-hops as the alternate next-hop(s) for that primary next hop or to use the MRT-Red next-hops. The algorithm is given in Figure 23 and discussed afterwards.

```

Select_Alternates_Internal(S, D, F, primary_intf,
                          D_lower, D_higher, D_topo_order)

    //When D==F, we can do only link protection
    if ((D is F) or (D.order_proxy is F))

```

```
    if an MRT doesn't use primary_intf
        indicate alternate is not node-protecting
        return that MRT color
    else // parallel links are cut-edge
        return AVOID_LINK_ON_BLUE

if (D_lower and D_higher and F_lower and F_higher)
    if F_topo_order < D_topo_order
        return USE_RED
    else
        return USE_BLUE

if (D_lower and D_higher)
    if F_higher
        return USE_RED
    else
        return USE_BLUE

if (F_lower and F_higher)
    if D_lower
        return USE_RED
    else if D_higher
        return USE_BLUE
    else
        if primary_intf.OUTGOING and primary_intf.INCOMING
            return AVOID_LINK_ON_BLUE
        if primary_intf.OUTGOING is true
            return USE_BLUE
        if primary_intf.INCOMING is true
            return USE_RED

if D_higher
    if F_higher
        if F_topo_order < D_topo_order
            return USE_RED
        else
            return USE_BLUE
    else if F_lower
        return USE_BLUE
    else
        // F and S are neighbors so either F << S or F >> S
else if D_lower
    if F_higher
        return USE_RED
    else if F_lower
        if F_topo_order < D_topo_order
            return USE_RED
        else
```

```

        return USE_BLUE
    else
        // F and S are neighbors so either F << S or F >> S
    else // D and S not ordered
        if F.lower
            return USE_RED
        else if F.higher
            return USE_BLUE
        else
            // F and S are neighbors so either F << S or F >> S

Select_Alternates(S, D, F, primary_intf)
    if D.order_proxy is not D
        D_lower = D.order_proxy.lower
        D_higher = D.order_proxy.higher
        D_topo_order = D.order_proxy.topo_order
    else
        D_lower = D.lower
        D_higher = D.higher
        D_topo_order = D.topo_order
    return Select_Alternates_Internal(S, D, F, primary_intf,
                                      D_lower, D_higher, D_topo_order)

```

Figure 23

If either $D \gg S \gg F$ or $D \ll S \ll F$ holds true, the situation is simple: in the first case we should choose the increasing Blue next-hop, in the second case, the decreasing Red next-hop is the right choice.

However, when both D and F are greater than S the situation is not so simple, there can be three possibilities: (i) $F \gg D$ (ii) $F \ll D$ or (iii) F and D are not ordered. In the first case, we should choose the path towards D along the Blue tree. In contrast, in case (ii) the Red path towards the root and then to D would be the solution. Finally, in case (iii) both paths would be acceptable. However, observe that if e.g. $F.topo_order > D.topo_order$, either case (i) or case (iii) holds true, which means that selecting the Blue next-hop is safe. Similarly, if $F.topo_order < D.topo_order$, we should select the Red next-hop. The situation is almost the same if both F and D are less than S.

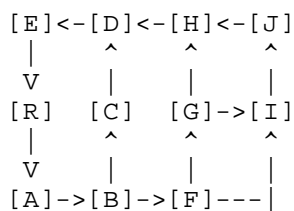
Recall that we have added each link to the GADAG in some direction, so it is impossible that S and F are not ordered. But it is possible that S and D are not ordered, so we need to deal with this case as well. If $F < S$, we can use the Red next-hop, because that path is first increasing until a node definitely greater than D is reached, then decreasing; this path must avoid using F. Similarly, if $F > S$, we should use the Blue next-hop.

Additionally, the cases where either F or D is ordered both higher and lower must be considered; this can happen when one is a block-root or its order_proxy is. If D is both higher and lower than S, then the MRT to use is the one that avoids F so if F is higher, then the MRT-Red should be used and if F is lower, then the MRT-Blue should be used; F and S must be ordered because they are neighbors. If F is both higher and lower, then if D is lower, using the MRT-Red to decrease reaches D and if D is higher, using the Blue MRT to increase reaches D; if D is unordered compared to S, then the situation is a bit more complicated.

In the case where $F < S < F$ and D and S are unordered, the direction of the link in the GADAG between S and F should be examined. If the link is directed $S \rightarrow F$, then use the MRT-Blue (decrease to avoid that link and then increase). If the link is directed $S \leftarrow F$, then use the MRT-Red (increase to avoid that link and then decrease). If the link is $S \leftrightarrow F$, then the link must be a cut-link and there is no node-protecting alternate. If there are multiple links between S and F, then they can protect against each other; of course, in this situation, they are probably already ECMP.

Finally, there is the case where D is also F. In this case, only link protection is possible. The MRT that doesn't use the indicated primary next-hop is used. If both MRTs use the primary next-hop, then the primary next-hop must be a cut-edge so either MRT could be used but the set of MRT next-hops must be pruned to avoid that primary next-hop. To indicate this case, `Select_Alternates` returns `AVOID_LINK_ON_BLUE`.

As an example, consider the ADAG depicted in Figure 24 and first suppose that G is the source, D is the destination and H is the failed next-hop. Since $D > G$, we need to compare `H.topo_order` and `D.topo_order`. Since $D.topo_order > H.topo_order$, D must be not smaller than H, so we should select the decreasing path towards the root. If, however, the destination were instead J, we must find that $H.topo_order > J.topo_order$, so we must choose the increasing Blue next-hop to J, which is I. In the case, when instead the destination is C, we find that we need to first decrease to avoid using H, so the Blue, first decreasing then increasing, path is selected.



(a)

a 2-connected graph

Figure 24

4.8. Finding FRR Next-Hops for Proxy-Nodes

As discussed in Section 10.2 of [I-D.ietf-rtgwg-mrt-frr-architecture], it is necessary to find MRT-Blue and MRT-Red next-hops and MRT-FRR alternates for a named proxy-nodes. An example case is for a router that is not part of that local MRT Island, when there is only partial MRT support in the domain.

A first incorrect and naive approach to handling proxy-nodes, which cannot be transited, is to simply add these proxy-nodes to the graph of the network and connect it to the routers through which the new proxy-node can be reached. Unfortunately, this can introduce some new ordering between the border routers connected to the new node which could result in routing MRT paths through the proxy-node. Thus, this naive approach would need to recompute GADAGs and redo SPTs for each proxy-node.

Instead of adding the proxy-node to the original network graph, each individual proxy-node can be individually added to the GADAG. The proxy-node is connected to at most two nodes in the GADAG. Section 10.2 of [I-D.ietf-rtgwg-mrt-frr-architecture] defines how the proxy-node attachments MUST be determined. The degenerate case where the proxy-node is attached to only one node in the GADAG is trivial as all needed information can be derived from that attachment node; if there are different interfaces, then some can be assigned to MRT-Red and others to MRT-Blue.

Now, consider the proxy-node that is attached to exactly two nodes in the GADAG. Let the `order_proxies` of these nodes be A and B. Let the current node, where next-hop is just being calculated, be S. If one of these two nodes A and B is the local root of S, let `A=S.local_root` and the other one be B. Otherwise, let `A.topo_order < B.topo_order`.

A valid GADAG was constructed. Instead doing an increasing-SPF and a decreasing-SPF to find ordering for the proxy-nodes, the following simple rules, providing the same result, can be used independently for each different proxy-node. For the following rules, let $X=A.local_root$, and if A is the local root, let that be strictly lower than any other node. Always take the first rule that matches.

Rule	Condition	Blue NH	Red NH	Notes
1	$S=X$	Blue to A	Red to B	
2	$S<<A$	Blue to A	Red to R	
3	$S>>B$	Blue to R	Red to B	
4	$A<<S<<B$	Red to A	Blue to B	
5	$A<<S$	Red to A	Blue to R	S not ordered w/ B
6	$S<<B$	Red to R	Blue to B	S not ordered w/ A
7	Otherwise	Red to R	Blue to R	S not ordered w/ A+B

These rules are realized in the following pseudocode where P is the proxy-node, X and Y are the nodes that P is attached to, and S is the computing router:

```

Select_Proxy_Node_NHs(P, S, X, Y)
  if (X.order_proxy.topo_order < Y.order_proxy.topo_order)
    //This fits even if X.order_proxy=S.local_root
    A=X.order_proxy
    B=Y.order_proxy
  else
    A=Y.order_proxy
    B=X.order_proxy

  if (S==A.local_root)
    P.blue_next_hops = A.blue_next_hops
    P.red_next_hops  = B.red_next_hops
    return
  if (A.higher)
    P.blue_next_hops = A.blue_next_hops
    P.red_next_hops  = R.red_next_hops
    return
  if (B.lower)
    P.blue_next_hops = R.blue_next_hops
    P.red_next_hops  = B.red_next_hops
    return
  if (A.lower && B.higher)
    P.blue_next_hops = A.red_next_hops
    P.red_next_hops  = B.blue_next_hops
    return
  if (A.lower)

```

```

        P.blue_next_hops = R.red_next_hops
        P.red_next_hops  = B.blue_next_hops
        return
    if (B.higher)
        P.blue_next_hops = A.red_next_hops
        P.red_next_hops  = R.blue_next_hops
        return
    P.blue_next_hops = R.red_next_hops
    P.red_next_hops  = R.blue_next_hops
    return

```

After finding the the red and the blue next-hops, it is necessary to know which one of these to use in the case of failure. This can be done by `Select_Alternates_Inner()`. In order to use `Select_Alternates_Internal()`, we need to know if P is greater, less or unordered with S, and P.topo_order. P.lower = B.lower, P.higher = A.higher, and any value is OK for P.topo_order, until A.topo_order<=P.topo_order<=B.topo_order and P.topo_order is not equal to the topo_order of the failed node. So for simplicity let P.topo_order=A.topo_order when the next-hop is not A, and P.topo_order=B.topo_order otherwise. This gives the following pseudo-code:

```

Select_Alternates_Proxy_Node(S, P, F, primary_intf)
    if (F is not P.neighbor_A)
        return Select_Alternates_Internal(S, P, F, primary_intf,
                                           P.neighbor_B.lower,
                                           P.neighbor_A.higher,
                                           P.neighbor_A.topo_order)
    else
        return Select_Alternates_Internal(S, P, F, primary_intf,
                                           P.neighbor_B.lower,
                                           P.neighbor_A.higher,
                                           P.neighbor_B.topo_order)

```

Figure 25

5. MRT Lowpoint Algorithm: Complete Specification

This specification defines the MRT Lowpoint Algorithm, which include the construction of a common GADAG and the computation of MRT-Red and MRT-Blue next-hops to each node in the graph. An implementation MAY select any subset of next-hops for MRT-Red and MRT-Blue that respect the available nodes that are described in Section 4.6 for each of the MRT-Red and MRT-Blue and the selected next-hops are further along in the interval of allowed nodes towards the destination.

For example, the MRT-Blue next-hops used when the destination $Y \gg S$, the computing router, MUST be one or more nodes, T , whose `topo_order` is in the interval $[X.topo_order, Y.topo_order]$ and where $Y \gg T$ or Y is T . Similarly, the MRT-Red next-hops MUST be have a `topo_order` in the interval $[R-small.topo_order, X.topo_order]$ or $[Y.topo_order, R-big.topo_order]$.

Implementations SHOULD implement the `Select_Alternates()` function to pick an MRT-FRR alternate.

In a future version, this section will include pseudo-code describing the full code path through the pseudo-code given earlier in the draft.

6. Algorithm Alternatives and Evaluation

This specification defines the MRT Lowpoint Algorithm, which is one option among several possible MRT algorithms. Other alternatives are described in the appendices.

In addition, it is possible to calculate Destination-Rooted GADAG, where for each destination, a GADAG rooted at that destination is computed. Then a router can compute the blue MRT and red MRT next-hops to that destination. Building GADAGs per destination is computationally more expensive, but may give somewhat shorter alternate paths. It may be useful for live-live multicast along MRTs.

6.1. Algorithm Evaluation

This section compares MRT and remote LFA for IP Fast Reroute in 19 service provider network topologies, focusing on coverage and alternate path length. Figure 26 shows the node-protecting coverage provided by local LFA (LLFA), remote LFA (RLFA), and MRT against different failure scenarios in these topologies. The coverage values are calculated as the percentage of source-destination pairs protected by the given IPFRR method relative to those protectable by optimal routing, against the same failure modes. More details on alternate selection policies used for this analysis are described later in this section.

Topology	percentage of failure scenarios covered by IPFRR method		
	NP_LLFA	NP_RLFA	MRT
T201	37	90	100
T202	73	83	100
T203	51	80	100
T204	55	81	100
T205	92	93	100
T206	71	74	100
T207	57	74	100
T208	66	81	100
T209	79	79	100
T210	95	98	100
T211	68	71	100
T212	59	63	100
T213	84	84	100
T214	68	78	100
T215	84	88	100
T216	43	59	100
T217	78	88	100
T218	72	75	100
T219	78	84	100

Figure 26

For the topologies analyzed here, LLFA is able to provide node-protecting coverage ranging from 37% to 95% of the source-destination pairs, as seen in the column labeled NP_LLFA. The use of RLFA in addition to LLFA is generally able to increase the node-protecting coverage. The percentage of node-protecting coverage with RLFA is provided in the column labeled NP_RLFA, ranges from 59% to 98% for these topologies. The node-protecting coverage provided by MRT is 100% since MRT is able to provide protection for any source-destination pair for which a path still exists after the failure.

We would also like to measure the quality of the alternate paths produced by these different IPFRR methods. An obvious approach is to take an average of the alternate path costs over all source-destination pairs and failure modes. However, this presents a problem, which we will illustrate by presenting an example of results for one topology using this approach (Figure 27). In this table, the average relative path length is the alternate path length for the IPFRR method divided by the optimal alternate path length, averaged

over all source-destination pairs and failure modes. The first three columns of data in the table give the path length calculated from the sum of IGP metrics of the links in the path. The results for topology T208 show that the metric-based path lengths for NP_LLFA and NP_RLFA alternates are on average 78 and 66 times longer than the path lengths for optimal alternates. The metric-based path lengths for MRT alternates are on average 14 times longer than for optimal alternates.

Topology	average relative alternate path length					
	IGP metric			hopcount		
	NP_LLFA	NP_RLFA	MRT	NP_LLFA	NP_RLFA	MRT
T208	78.2	66.0	13.6	0.99	1.01	1.32

Figure 27

The network topology represented by T208 uses values of 10, 100, and 1000 as IGP costs, so small deviations from the optimal alternate path can result in large differences in relative path length. LLFA, RLFA, and MRT all allow for at least one hop in the alternate path to be chosen independent of the cost of the link. This can easily result in an alternate using a link with cost 1000, which introduces noise into the path length measurement. In the case of T208, the adverse effects of using metric-based path lengths is obvious. However, we have observed that the metric-based path length introduces noise into alternate path length measurements in several other topologies as well. For this reason, we have opted to measure the alternate path length using hopcount. While IGP metrics may be adjusted by the network operator for a number of reasons (e.g. traffic engineering), the hopcount is a fairly stable measurement of path length. As shown in the last three columns of Figure 27, the hopcount-based alternate path lengths for topology T208 are fairly well-behaved.

Figure 28, Figure 29, Figure 30, and Figure 31 present the hopcount-based path length results for the 19 topologies examined. The topologies in the four tables are grouped based on the size of the topologies, as measured by the number of nodes, with Figure 28 having the smallest topologies and Figure 31 having the largest topologies. Instead of trying to represent the path lengths of a large set of alternates with a single number, we have chosen to present a histogram of the path lengths for each IPFRR method and alternate selection policy studied. The first eight columns of data represent

the percentage of failure scenarios protected by an alternate N hops longer than the primary path, with the first column representing an alternate 0 or 1 hops longer than the primary path, all the way up through the eighth column representing an alternate 14 or 15 hops longer than the primary path. The last column in the table gives the percentage of failure scenarios for which there is no alternate less than 16 hops longer than the primary path. In the case of LLFA and RLFA, this category includes failure scenarios for which no alternate was found.

For each topology, the first row (labeled OPTIMAL) is the distribution of the number of hops in excess of the primary path hopcount for optimally routed alternates. (The optimal routing was done with respect to IGP metrics, as opposed to hopcount.) The second row (labeled NP_LLFA) is the distribution of the extra hops for node-protecting LLFA. The third row (labeled NP_LLFA_THEN_NP_RLFA) is the hopcount distribution when one adds node-protecting RLFA to increase the coverage. The alternate selection policy used here first tries to find a node-protecting LLFA. If that does not exist, then it tries to find an RLFA, and checks if it is node-protecting. Comparing the hopcount distribution for RLFA and LLFA across these topologies, one can see how the coverage is increased at the expense of using longer alternates. It is also worth noting that while superficially LLFA and RLFA appear to have better hopcount distributions than OPTIMAL, the presence of entries in the last column (no alternate < 16) mainly represent failure scenarios that are not protected, for which the hopcount is effectively infinite.

The fourth and fifth rows of each topology show the hopcount distributions for two alternate selection policies using MRT alternates. The policy represented by the label NP_LLFA_THEN_MRT_LOWPOINT will first use a node-protecting LLFA. If a node-protecting LLFA does not exist, then it will use an MRT alternate. The policy represented by the label MRT_LOWPOINT instead will use the MRT alternate even if a node-protecting LLFA exists. One can see from the data that combining node-protecting LLFA with MRT results in a significant shortening of the alternate hopcount distribution.

Topology name and alternate selection policy evaluated	percentage of failure scenarios protected by an alternate N hops longer than the primary path									
						10	12	14	no	
	0-1	2-3	4-5	6-7	8-9	-11	-13	-15	alt <16	
T201(avg primary hops=3.5)										
OPTIMAL	37	37	20	3	3				63 10	
NP_LLFA	37									
NP_LLFA_THEN_NP_RLFA	37	34	19							
NP_LLFA_THEN_MRT_LOWPOINT	37	33	21	6	3					
MRT_LOWPOINT	33	36	23	6	3					
T202(avg primary hops=4.8)										
OPTIMAL	90	9							27 17	
NP_LLFA	71	2								
NP_LLFA_THEN_NP_RLFA	78	5								
NP_LLFA_THEN_MRT_LOWPOINT	80	12	5	2	1					
MRT_LOWPOINT_ONLY	48	29	13	7	2	1				
T203(avg primary hops=4.1)										
OPTIMAL	36	37	21	4	2				49 20	
NP_LLFA	34	15	3							
NP_LLFA_THEN_NP_RLFA	35	19	22	4						
NP_LLFA_THEN_MRT_LOWPOINT	36	35	22	5	2					
MRT_LOWPOINT_ONLY	31	35	26	7	2					
T204(avg primary hops=3.7)										
OPTIMAL	76	20	3	1					45 19	
NP_LLFA	54	1								
NP_LLFA_THEN_NP_RLFA	67	10	4							
NP_LLFA_THEN_MRT_LOWPOINT	70	18	8	3	1					
MRT_LOWPOINT_ONLY	58	27	11	3	1					
T205(avg primary hops=3.4)										
OPTIMAL	92	8							8 7	
NP_LLFA	89	3								
NP_LLFA_THEN_NP_RLFA	90	4								
NP_LLFA_THEN_MRT_LOWPOINT	91	9								
MRT_LOWPOINT_ONLY	62	33	5	1						

Figure 28

Topology name and alternate selection policy evaluated	percentage of failure scenarios protected by an alternate N hops longer than the primary path								
	0-1	2-3	4-5	6-7	8-9	10-11	12-13	14-15	no alt <16
T206(avg primary hops=3.7)									
OPTIMAL	63	30	7						
NP_LLFA	60	9	1						29
NP_LLFA_THEN_NP_RLFA	60	13	1						26
NP_LLFA_THEN_MRT_LOWPOINT	64	29	7						
MRT_LOWPOINT	55	32	13						
T207(avg primary hops=3.9)									
OPTIMAL	71	24	5	1					
NP_LLFA	55	2							43
NP_LLFA_THEN_NP_RLFA	63	10							26
NP_LLFA_THEN_MRT_LOWPOINT	70	20	7	2	1				
MRT_LOWPOINT_ONLY	57	29	11	3	1				
T208(avg primary hops=4.6)									
OPTIMAL	58	28	12	2	1				
NP_LLFA	53	11	3						34
NP_LLFA_THEN_NP_RLFA	56	17	7	1					19
NP_LLFA_THEN_MRT_LOWPOINT	58	19	10	7	3	1			
MRT_LOWPOINT_ONLY	34	24	21	13	6	2	1		
T209(avg primary hops=3.6)									
OPTIMAL	85	14	1						
NP_LLFA	79								21
NP_LLFA_THEN_NP_RLFA	79								21
NP_LLFA_THEN_MRT_LOWPOINT	82	15	2						
MRT_LOWPOINT_ONLY	63	29	8						
T210(avg primary hops=2.5)									
OPTIMAL	95	4	1						
NP_LLFA	94	1							5
NP_LLFA_THEN_NP_RLFA	94	3	1						2
NP_LLFA_THEN_MRT_LOWPOINT	95	4	1						
MRT_LOWPOINT_ONLY	91	6	2						

Figure 29

Topology name and alternate selection policy evaluated	percentage of failure scenarios protected by an alternate N hops longer than the primary path									
	0-1	2-3	4-5	6-7	8-9	10-11	12-13	14-15	no alt	<16
T211(avg primary hops=3.3)										
OPTIMAL	88	11								
NP_LLFA	66	1								32
NP_LLFA_THEN_NP_RLFA	68	3								29
NP_LLFA_THEN_MRT_LOWPOINT	88	12								
MRT_LOWPOINT	85	15	1							
T212(avg primary hops=3.5)										
OPTIMAL	76	23	1							
NP_LLFA	59									41
NP_LLFA_THEN_NP_RLFA	61	1	1							37
NP_LLFA_THEN_MRT_LOWPOINT	75	24	1							
MRT_LOWPOINT_ONLY	66	31	3							
T213(avg primary hops=4.3)										
OPTIMAL	91	9								
NP_LLFA	84									16
NP_LLFA_THEN_NP_RLFA	84									16
NP_LLFA_THEN_MRT_LOWPOINT	89	10	1							
MRT_LOWPOINT_ONLY	75	24	1							
T214(avg primary hops=5.8)										
OPTIMAL	71	22	5	2						
NP_LLFA	58	8	1	1						32
NP_LLFA_THEN_NP_RLFA	61	13	3	1						22
NP_LLFA_THEN_MRT_LOWPOINT	66	14	7	5	3	2	1	1	1	1
MRT_LOWPOINT_ONLY	30	20	18	12	8	4	3	2	3	3
T215(avg primary hops=4.8)										
OPTIMAL	73	27								
NP_LLFA	73	11								16
NP_LLFA_THEN_NP_RLFA	73	13	2							12
NP_LLFA_THEN_MRT_LOWPOINT	74	19	3	2	1	1	1			
MRT_LOWPOINT_ONLY	32	31	16	12	4	3	1			

Figure 30

Topology name and alternate selection policy evaluated	percentage of failure scenarios protected by an alternate N hops longer than the primary path								
	0-1	2-3	4-5	6-7	8-9	10-11	12-13	14-15	no alt <16
T216(avg primary hops=5.2)									
OPTIMAL	60	32	7	1					
NP_LLFA	39	4							57
NP_LLFA_THEN_NP_RLFA	46	12	2						41
NP_LLFA_THEN_MRT_LOWPOINT	48	20	12	7	5	4	2	1	1
MRT_LOWPOINT	28	25	18	11	7	6	3	2	1
T217(avg primary hops=8.0)									
OPTIMAL	81	13	5	1					
NP_LLFA	74	3	1						22
NP_LLFA_THEN_NP_RLFA	76	8	3	1					12
NP_LLFA_THEN_MRT_LOWPOINT	77	7	5	4	3	2	1	1	
MRT_LOWPOINT_ONLY	25	18	18	16	12	6	3	1	
T218(avg primary hops=5.5)									
OPTIMAL	85	14	1						
NP_LLFA	68	3							28
NP_LLFA_THEN_NP_RLFA	71	4							25
NP_LLFA_THEN_MRT_LOWPOINT	77	12	7	4	1				
MRT_LOWPOINT_ONLY	37	29	21	10	3	1			
T219(avg primary hops=7.7)									
OPTIMAL	77	15	5	1	1				
NP_LLFA	72	5							22
NP_LLFA_THEN_NP_RLFA	73	8	2						16
NP_LLFA_THEN_MRT_LOWPOINT	74	8	3	3	2	2	2	2	4
MRT_LOWPOINT_ONLY	19	14	15	12	10	8	7	6	10

Figure 31

In the preceding analysis, the following procedure for selecting an RLFA was used. Nodes were ordered with respect to distance from the source and checked for membership in Q and P-space. The first node to satisfy this condition was selected as the RLFA. More sophisticated methods to select node-protecting RLFAs is an area of active research.

The analysis presented above uses the MRT Lowpoint Algorithm defined in this specification with a common GADAG root. The particular choice of a common GADAG root is expected to affect the quality of the MRT alternate paths, with a more central common GADAG root resulting in shorter MRT alternate path lengths. For the analysis above, the GADAG root was chosen for each topology by calculating node centrality as the sum of costs of all shortest paths to and from a given node. The node with the lowest sum was chosen as the common GADAG root. In actual deployments, the common GADAG root would be chosen based on the GADAG Root Selection Priority advertised by each router, the values of which would be determined off-line.

In order to measure how sensitive the MRT alternate path lengths are to the choice of common GADAG root, we performed the same analysis using different choices of GADAG root. All of the nodes in the network were ordered with respect to the node centrality as computed above. Nodes were chosen at the 0th, 25th, and 50th percentile with respect to the centrality ordering, with 0th percentile being the most central node. The distribution of alternate path lengths for those three choices of GADAG root are shown in Figure 32 for a subset of the 19 topologies (chosen arbitrarily). The third row for each topology (labeled MRT_LOWPOINT (0 percentile)) reproduces the results presented above for MRT_LOWPOINT_ONLY. The fourth and fifth rows show the alternate path length distribution for the 25th and 50th percentile choice for GADAG root. One can see some impact on the path length distribution with the less central choice of GADAG root resulting in longer path lengths.

We also looked at the impact of MRT algorithm variant on the alternate path lengths. The first two rows for each topology present results of the same alternate path length distribution analysis for the SPF and Hybrid methods for computing the GADAG. These two methods are described in Appendix A and Appendix B. For three of the topologies in this subset (T201, T206, and T211), the use of SPF or Hybrid methods does not appear to provide a significant advantage over the Lowpoint method with respect to path length. Instead, the choice of GADAG root appears to have more impact on the path length. However, for two of the topologies in this subset (T216 and T219) and for this particular choice of GADAG root, the use of the SPF method results in noticeably shorter alternate path lengths than the use of the Lowpoint or Hybrid methods. It remains to be determined if this effect applies generally across more topologies or is sensitive to choice of GADAG root.

Topology name	percentage of failure scenarios protected by an alternate N hops longer than the primary path									
MRT algorithm variant										
(GADAG root centrality percentile)	0-1	2-3	4-5	6-7	8-9	10-11	12-13	14-15	no alt	<16
T201(avg primary hops=3.5)										
MRT_HYBRID (0 percentile)	33	26	23	6	3					
MRT_SPF (0 percentile)	33	36	23	6	3					
MRT_LOWPOINT (0 percentile)	33	36	23	6	3					
MRT_LOWPOINT (25 percentile)	27	29	23	11	10					
MRT_LOWPOINT (50 percentile)	27	29	23	11	10					
T206(avg primary hops=3.7)										
MRT_HYBRID (0 percentile)	50	35	13	2						
MRT_SPF (0 percentile)	50	35	13	2						
MRT_LOWPOINT (0 percentile)	55	32	13							
MRT_LOWPOINT (25 percentile)	47	25	22	6						
MRT_LOWPOINT (50 percentile)	38	38	14	11						
T211(avg primary hops=3.3)										
MRT_HYBRID (0 percentile)	86	14								
MRT_SPF (0 percentile)	86	14								
MRT_LOWPOINT (0 percentile)	85	15	1							
MRT_LOWPOINT (25 percentile)	70	25	5	1						
MRT_LOWPOINT (50 percentile)	80	18	2							
T216(avg primary hops=5.2)										
MRT_HYBRID (0 percentile)	23	22	18	13	10	7	4	2	2	
MRT_SPF (0 percentile)	35	32	19	9	3	1				
MRT_LOWPOINT (0 percentile)	28	25	18	11	7	6	3	2	1	
MRT_LOWPOINT (25 percentile)	24	20	19	16	10	6	3	1		
MRT_LOWPOINT (50 percentile)	19	14	13	10	8	6	5	5	10	
T219(avg primary hops=7.7)										
MRT_HYBRID (0 percentile)	20	16	13	10	7	5	5	5	3	
MRT_SPF (0 percentile)	31	23	19	12	7	4	2	1		
MRT_LOWPOINT (0 percentile)	19	14	15	12	10	8	7	6	10	
MRT_LOWPOINT (25 percentile)	19	14	15	13	12	10	6	5	7	
MRT_LOWPOINT (50 percentile)	19	14	14	12	11	8	6	6	10	

Figure 32

7. Algorithm Work to Be Done

Broadcast Interfaces: The algorithm assumes that broadcast interfaces are already represented as pseudo-nodes in the network graph. Given maximal redundancy, one of the MRT will try to avoid both the pseudo-node and the next hop. The exact rules need to be fully specified.

8. IANA Considerations

This document includes no request to IANA.

9. Security Considerations

This architecture is not currently believed to introduce new security concerns.

10. References

10.1. Normative References

[I-D.ietf-rtgwg-mrt-frr-architecture]
Atlas, A., Kebler, R., Envedi, G., Csaszar, A., Tantsura, J., Konstantynowicz, M., and R. White, "An Architecture for IP/LDP Fast-Reroute Using Maximally Redundant Trees", draft-ietf-rtgwg-mrt-frr-architecture-03 (work in progress), July 2013.

10.2. Informative References

[EnyediThesis]
Enyedi, G., "Novel Algorithms for IP Fast Reroute", Department of Telecommunications and Media Informatics, Budapest University of Technology and Economics Ph.D. Thesis, February 2011, <http://www.omikk.bme.hu/collections/phd/Villamosmernoki_es_Informatikai_Kar/2011/Enyedi_Gabor/ertekezes.pdf>.

[I-D.atlas-ospf-mrt]
Atlas, A., Hegde, S., Chris, C., and J. Tantsura, "OSPF Extensions to Support Maximally Redundant Trees", draft-atlas-ospf-mrt-00 (work in progress), July 2013.

[I-D.ietf-rtgwg-ipfrr-notvia-addresses]
Bryant, S., Previdi, S., and M. Shand, "A Framework for IP and MPLS Fast Reroute Using Not-via Addresses", draft-ietf-rtgwg-ipfrr-notvia-addresses-11 (work in progress), May 2013.

- [I-D.ietf-rtgwg-lfa-manageability]
Litkowski, S., Decraene, B., Filsfils, C., and K. Raza,
"Operational management of Loop Free Alternates", draft-
ietf-rtgwg-lfa-manageability-00 (work in progress), May
2013.
- [I-D.ietf-rtgwg-remote-lfa]
Bryant, S., Filsfils, C., Previdi, S., Shand, M., and S.
Ning, "Remote LFA FRR", draft-ietf-rtgwg-remote-lfa-02
(work in progress), May 2013.
- [Kahn_1962_topo_sort]
Kahn, A., "Topological sorting of large networks",
Communications of the ACM, Volume 5, Issue 11 , Nov 1962,
<<http://dl.acm.org/citation.cfm?doid=368996.369025>>.
- [LFARevisited]
Retvari, G., Tapolcai, J., Enyedi, G., and A. Csaszar, "IP
Fast ReRoute: Loop Free Alternates Revisited", Proceedings
of IEEE INFOCOM , 2011, <http://opti.tmit.bme.hu/~tapolcai/papers/retvari2011lfa_infocom.pdf>.
- [LightweightNotVia]
Enyedi, G., Retvari, G., Szilagyi, P., and A. Csaszar, "IP
Fast ReRoute: Lightweight Not-Via without Additional
Addresses", Proceedings of IEEE INFOCOM , 2009,
<<http://mycite.omikk.bme.hu/doc/71691.pdf>>.
- [MRTLlinear]
Enyedi, G., Retvari, G., and A. Csaszar, "On Finding
Maximally Redundant Trees in Strictly Linear Time", IEEE
Symposium on Computers and Communications (ISCC) , 2009,
<<http://opti.tmit.bme.hu/~enyedi/ipfrr/distMaxRedTree.pdf>>.
- [RFC3137] Retana, A., Nguyen, L., White, R., Zinin, A., and D.
McPherson, "OSPF Stub Router Advertisement", RFC 3137,
June 2001.
- [RFC5286] Atlas, A. and A. Zinin, "Basic Specification for IP Fast
Reroute: Loop-Free Alternates", RFC 5286, September 2008.
- [RFC5714] Shand, M. and S. Bryant, "IP Fast Reroute Framework", RFC
5714, January 2010.

[RFC6571] Filsfils, C., Francois, P., Shand, M., Decraene, B., Uttaro, J., Leymann, N., and M. Horneffer, "Loop-Free Alternate (LFA) Applicability in Service Provider (SP) Networks", RFC 6571, June 2012.

Appendix A. Option 2: Computing GADAG using SPF

The basic idea in this option is to use slightly-modified SPF computations to find ears. In every block, an SPF computation is first done to find a cycle from the local root and then SPF computations in that block find ears until there are no more interfaces to be explored. The used result from the SPF computation is the path of interfaces indicated by following the previous hops from the minimized IN_GADAG node back to the SPF root.

To do this, first all cut-vertices must be identified and local-roots assigned as specified in Figure 12.

The slight modifications to the SPF are as follows. The root of the block is referred to as the block-root; it is either the GADAG root or a cut-vertex.

- a. The SPF is rooted at a neighbor x of an IN_GADAG node y. All links between y and x are marked as TEMP_UNUSABLE. They should not be used during the SPF computation.
- b. If y is not the block-root, then it is marked TEMP_UNUSABLE. It should not be used during the SPF computation. This prevents ears from starting and ending at the same node and avoids cycles; the exception is because cycles to/from the block-root are acceptable and expected.
- c. Do not explore links to nodes whose local-root is not the block-root. This keeps the SPF confined to the particular block.
- d. Terminate when the first IN_GADAG node z is minimized.
- e. Respect the existing directions (e.g. INCOMING, OUTGOING, UNDIRECTED) already specified for each interface.

```
Mod_SPF(spf_root, block_root)
  Initialize spf_heap to empty
  Initialize nodes' spf_metric to infinity
  spf_root.spf_metric = 0
  insert(spf_heap, spf_root)
  found_in_gadag = false
  while (spf_heap is not empty) and (found_in_gadag is false)
```



```

min_node = remove_lowest(spf_heap)
if min_node.IN_GADAG is true
    found_in_gadag = true
else
    foreach interface intf of min_node
        if ((intf.OUTGOING or intf.UNDIRECTED) and
            ((intf.remote_node.localroot is block_root) or
             (intf.remote_node is block_root)) and
            (intf.remote_node is not TEMP_UNUSABLE) and
            (intf is not TEMP_UNUSABLE))
            path_metric = min_node.spf_metric + intf.metric
            if path_metric < intf.remote_node.spf_metric
                intf.remote_node.spf_metric = path_metric
                intf.remote_node.spf_prev_intf = intf
                insert_or_update(spf_heap, intf.remote_node)
    return min_node

SPF_for_Ear(cand_intf.local_node, cand_intf.remote_node, block_root,
            method)
    Mark all interfaces between cand_intf.remote_node
        and cand_intf.local_node as TEMP_UNUSABLE
    if cand_intf.local_node is not block_root
        Mark cand_intf.local_node as TEMP_UNUSABLE
    Initialize ear_list to empty
    end_ear = Mod_SPF(spf_root, block_root)
    y = end_ear.spf_prev_hop
    while y.local_node is not spf_root
        add_to_list_start(ear_list, y)
        y.local_node.IN_GADAG = true
        y = y.local_node.spf_prev_intf
    if(method is not hybrid)
        Set_Ear_Direction(ear_list, cand_intf.local_node,
                           end_ear, block_root)
    Clear TEMP_UNUSABLE from all interfaces between
        cand_intf.remote_node and cand_intf.local_node
    Clear TEMP_UNUSABLE from cand_intf.local_node
    return end_ear

```

Figure 33: Modified SPF for GADAG computation

Assume that an ear is found by going from y to x and then running an SPF that terminates by minimizing z (e.g. $y \leftrightarrow x \dots q \leftrightarrow z$). Now it is necessary to determine the direction of the ear; if $y \ll z$, then the path should be $y \rightarrow x \dots q \rightarrow z$ but if $y \gg z$, then the path should be $y \leftarrow x \dots q \leftarrow z$. In Section 4.4, the same problem was handled by finding

all ears that started at a node before looking at ears starting at nodes higher in the partial order. In this algorithm, using that approach could mean that new ears aren't added in order of their total cost since all ears connected to a node would need to be found before additional nodes could be found.

The alternative is to track the order relationship of each node with respect to every other node. This can be accomplished by maintaining two sets of nodes at each node. The first set, `Higher_Nodes`, contains all nodes that are known to be ordered above the node. The second set, `Lower_Nodes`, contains all nodes that are known to be ordered below the node. This is the approach used in this algorithm.

```
Set_Ear_Direction(ear_list, end_a, end_b, block_root)
// Default of A_TO_B for the following cases:
// (a) end_a and end_b are the same (root)
// or (b) end_a is in end_b's Lower_Nodes
// or (c) end_a and end_b were unordered with respect to each
// other
direction = A_TO_B
if (end_b is block_root) and (end_a is not end_b)
    direction = B_TO_A
else if end_a is in end_b.Higher_Nodes
    direction = B_TO_A
if direction is B_TO_A
    foreach interface i in ear_list
        i.UNDIRECTED = false
        i.INCOMING = true
        i.remote_intf.UNDIRECTED = false
        i.remote_intf.OUTGOING = true
else
    foreach interface i in ear_list
        i.UNDIRECTED = false
        i.OUTGOING = true
        i.remote_intf.UNDIRECTED = false
        i.remote_intf.INCOMING = true
if end_a is end_b
    return
// Next, update all nodes' Lower_Nodes and Higher_Nodes
if (end_a is in end_b.Higher_Nodes)
    foreach node x where x.localroot is block_root
        if end_a is in x.Lower_Nodes
            foreach interface i in ear_list
                add i.remote_node to x.Lower_Nodes
        if end_b is in x.Higher_Nodes
            foreach interface i in ear_list
                add i.local_node to x.Higher_Nodes
```

```

else
  foreach node x where x.localroot is block_root
    if end_b is in x.Lower_Nodes
      foreach interface i in ear_list
        add i.local_node to x.Lower_Nodes
    if end_a is in x.Higher_Nodes
      foreach interface i in ear_list
        add i.remote_node to x.Higher_Nodes

```

Figure 34: Algorithm to assign links of an ear direction

A goal of the algorithm is to find the shortest cycles and ears. An ear is started by going to a neighbor x of an IN_GADAG node y. The path from x to an IN_GADAG node is minimal, since it is computed via SPF. Since a shortest path is made of shortest paths, to find the shortest ears requires reaching from the set of IN_GADAG nodes to the closest node that isn't IN_GADAG. Therefore, an ordered tree is maintained of interfaces that could be explored from the IN_GADAG nodes. The interfaces are ordered by their characteristics of metric, local loopback address, remote loopback address, and ifindex, as in the algorithm previously described in Figure 14.

The algorithm ignores interfaces picked from the ordered tree that belong to the block root if the block in which the interface is present already has an ear that has been computed. This is necessary since we allow at most one incoming interface to a block root in each block. This requirement stems from the way next-hops are computed as will be seen in Section 4.6. After any ear gets computed, we traverse the newly added nodes to the GADAG and insert interfaces whose far end is not yet on the GADAG to the ordered tree for later processing.

Finally, cut-edges are a special case because there is no point in doing an SPF on a block of 2 nodes. The algorithm identifies cut-edges simply as links where both ends of the link are cut-vertices. Cut-edges can simply be added to the GADAG with both OUTGOING and INCOMING specified on their interfaces.

```

add_eligible_interfaces_of_node(ordered_intfs_tree,node)
  for each interface of node
    if intf.remote_node.IN_GADAG is false
      insert(intf,ordered_intfs_tree)

check_if_block_has_ear(x,block_id)
  block_has_ear = false
  for all interfaces of x
    if (intf.remote_node.block_id == block_id) &&
      (intf.remote_node.IN_GADAG is true)

```

```

        block_has_ear = true
    return block_has_ear

Construct_GADAG_via_SPF(topology, root)
    Compute_Localroot (root,root)
    Assign_Block_ID(root,0)
    root.IN_GADAG = true
    add_eligible_interfaces_of_node(ordered_intfs_tree,root)
    while ordered_intfs_tree is not empty
        cand_intf = remove_lowest(ordered_intfs_tree)
        if cand_intf.remote_node.IN_GADAG is false
            if L(cand_intf.remote_node) == D(cand_intf.remote_node)
                // Special case for cut-edges
                cand_intf.UNDIRECTED = false
                cand_intf.remote_intf.UNDIRECTED = false
                cand_intf.OUTGOING = true
                cand_intf.INCOMING = true
                cand_intf.remote_intf.OUTGOING = true
                cand_intf.remote_intf.INCOMING = true
                cand_intf.remote_node.IN_GADAG = true
            add_eligible_interfaces_of_node(
                ordered_intfs_tree,cand_intf.remote_node)
        else
            if (cand_intf.remote_node.local_root ==
                cand_intf.local_node) &&
                check_if_block_has_ear
                    (cand_intf.local_node,
                     cand_intf.remote_node.block_id))
                /* Skip the interface since the block root
                   already has an incoming interface in the
                   block */
            else
                ear_end = SPF_for_Ear(cand_intf.local_node,
                                       cand_intf.remote_node,
                                       cand_intf.remote_node.localroot,
                                       SPF method)
                y = ear_end.spf_prev_hop
                while y.local_node is not cand_intf.local_node
                    add_eligible_interfaces_of_node(
                        ordered_intfs_tree,
                        y.local_node)
                    y = y.local_node.spf_prev_intf

```

Figure 35: SPF-based GADAG algorithm

Appendix B. Option 3: Computing GADAG using a hybrid method

In this option, the idea is to combine the salient features of the above two options. To this end, we process nodes as they get added to the GADAG just like in the lowpoint inheritance by maintaining a stack of nodes. This ensures that we do not need to maintain lower and higher sets at each node to ascertain ear directions since the ears will always be directed from the node being processed towards the end of the ear. To compute the ear however, we resort to an SPF to have the possibility of better ears (path lengths) thus giving more flexibility than the restricted use of lowpoint/dfs parents.

Regarding ears involving a block root, unlike the SPF method which ignored interfaces of the block root after the first ear, in the hybrid method we would have to process all interfaces of the block root before moving on to other nodes in the block since the direction of an ear is pre-determined. Thus, whenever the block already has an ear computed, and we are processing an interface of the block root, we mark the block root as unusable before the SPF run that computes the ear. This ensures that the SPF terminates at some node other than the block-root. This in turn guarantees that the block-root has only one incoming interface in each block, which is necessary for correctly computing the next-hops on the GADAG.

As in the SPF gadag, bridge ears are handled as a special case.

The entire algorithm is shown below in Figure 36

```

find_spf_stack_ear(stack, x, y, xy_intf, block_root)
  if L(y) == D(y)
    // Special case for cut-edges
    xy_intf.UNDIRECTED = false
    xy_intf.remote_intf.UNDIRECTED = false
    xy_intf.OUTGOING = true
    xy_intf.INCOMING = true
    xy_intf.remote_intf.OUTGOING = true
    xy_intf.remote_intf.INCOMING = true
    xy_intf.remote_node.IN_GADAG = true
    push y onto stack
    return
  else
    if (y.local_root == x) &&
      check_if_block_has_ear(x,y.block_id)
      //Avoid the block root during the SPF
      Mark x as TEMP_UNUSABLE
    end_ear = SPF_for_Ear(x,y,block_root,hybrid)
    If x was set as TEMP_UNUSABLE, clear it
    cur = end_ear
    while (cur != y)
      intf = cur.spf_prev_hop

```

```

        prev = intf.local_node
        intf.UNDIRECTED = false
        intf.remote_intf.UNDIRECTED = false
        intf.OUTGOING = true
        intf.remote_intf.INCOMING = true
        push prev onto stack
    cur = prev
    xy_intf.UNDIRECTED = false
    xy_intf.remote_intf.UNDIRECTED = false
    xy_intf.OUTGOING = true
    xy_intf.remote_intf.INCOMING = true
    return

Construct_GADAG_via_hybrid(topology,root)
Compute_Localroot (root,root)
Assign_Block_ID(root,0)
root.IN_GADAG = true
Initialize Stack to empty
push root onto Stack
while (Stack is not empty)
    x = pop(Stack)
    for each interface intf of x
        y = intf.remote_node
        if y.IN_GADAG is false
            find_spf_stack_ear(stack, x, y, intf, y.block_root)

```

Figure 36: Hybrid GADAG algorithm

Authors' Addresses

Gabor Sandor Enyedi (editor)
 Ericsson
 Konyves Kalman krt 11
 Budapest 1097
 Hungary

Email: Gabor.Sandor.Enyedi@ericsson.com

Andras Csaszar
 Ericsson
 Konyves Kalman krt 11
 Budapest 1097
 Hungary

Email: Andras.Csaszar@ericsson.com

Alia Atlas (editor)
Juniper Networks
10 Technology Park Drive
Westford, MA 01886
USA

Email: akatlas@juniper.net

Chris Bowers
Juniper Networks
1194 N. Mathilda Ave.
Sunnyvale, CA 94089
USA

Email: cbowers@juniper.net

Abishek Gopalan
University of Arizona
1230 E Speedway Blvd.
Tucson, AZ 85721
USA

Email: abishek@ece.arizona.edu

Routing Area Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 8, 2016

A. Atlas
C. Bowers
Juniper Networks
G. Enyedi
Ericsson
February 5, 2016

An Architecture for IP/LDP Fast-Reroute Using Maximally Redundant Trees
draft-ietf-rtgwg-mrt-frr-architecture-10

Abstract

This document defines the architecture for IP and LDP Fast-Reroute using Maximally Redundant Trees (MRT-FRR). MRT-FRR is a technology that gives link-protection and node-protection with 100% coverage in any network topology that is still connected after the failure.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on August 8, 2016.

Copyright Notice

Copyright (c) 2016 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of

the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
1.1. Importance of 100% Coverage	4
1.2. Partial Deployment and Backwards Compatibility	5
2. Requirements Language	5
3. Terminology	5
4. Maximally Redundant Trees (MRT)	7
5. Maximally Redundant Trees (MRT) and Fast-Reroute	9
6. Unicast Forwarding with MRT Fast-Reroute	9
6.1. Introduction to MRT Forwarding Options	10
6.1.1. MRT LDP labels	10
6.1.1.1. Topology-scoped FEC encoded using a single label (Option 1A)	10
6.1.1.2. Topology and FEC encoded using a two label stack (Option 1B)	11
6.1.1.3. Compatibility of MRT LDP Label Options 1A and 1B	12
6.1.1.4. Required support for MRT LDP Label options	12
6.1.2. MRT IP tunnels (Options 2A and 2B)	12
6.2. Forwarding LDP Unicast Traffic over MRT Paths	13
6.2.1. Forwarding LDP traffic using MRT LDP Label Option 1A	13
6.2.2. Forwarding LDP traffic using MRT LDP Label Option 1B	14
6.2.3. Other considerations for forwarding LDP traffic using MRT LDP Labels	14
6.2.4. Required support for LDP traffic	14
6.3. Forwarding IP Unicast Traffic over MRT Paths	14
6.3.1. Tunneling IP traffic using MRT LDP Labels	15
6.3.1.1. Tunneling IP traffic using MRT LDP Label Option 1A	15
6.3.1.2. Tunneling IP traffic using MRT LDP Label Option 1B	15
6.3.2. Tunneling IP traffic using MRT IP Tunnels	16
6.3.3. Required support for IP traffic	16
7. MRT Island Formation	16
7.1. IGP Area or Level	17
7.2. Support for a specific MRT profile	17
7.3. Excluding additional routers and interfaces from the MRT Island	18
7.3.1. Existing IGP exclusion mechanisms	18
7.3.2. MRT-specific exclusion mechanism	19
7.4. Connectivity	19
7.5. Algorithm for MRT Island Identification	19
8. MRT Profile	19
8.1. MRT Profile Options	19
8.2. Router-specific MRT paramaters	21

8.3. Default MRT profile	21
9. LDP signaling extensions and considerations	22
10. Inter-area Forwarding Behavior	22
10.1. ABR Forwarding Behavior with MRT LDP Label Option 1A . .	23
10.1.1. Motivation for Creating the Rainbow-FEC	24
10.2. ABR Forwarding Behavior with IP Tunneling (option 2) . .	24
10.3. ABR Forwarding Behavior with MRT LDP Label option 1B . .	25
11. Prefixes Multiply Attached to the MRT Island	26
11.1. Protecting Multi-Homed Prefixes using Tunnel Endpoint Selection	28
11.2. Protecting Multi-Homed Prefixes using Named Proxy-Nodes	29
11.3. MRT Alternates for Destinations Outside the MRT Island .	31
12. Network Convergence and Preparing for the Next Failure . . .	31
12.1. Micro-loop prevention and MRTs	32
12.2. MRT Recalculation for the Default MRT Profile	33
13. Implementation Status	34
14. Operational Considerations	35
14.1. Verifying Forwarding on MRT Paths	35
14.2. Traffic Capacity on Backup Paths	36
14.3. MRT IP Tunnel Loopback Address Management	38
14.4. MRT-FRR in a Network with Degraded Connectivity	38
14.5. Partial Deployment of MRT-FRR in a Network	38
15. Acknowledgements	39
16. IANA Considerations	39
17. Security Considerations	40
18. Contributors	40
19. References	41
19.1. Normative References	41
19.2. Informative References	42
Appendix A. Inter-level Forwarding Behavior for IS-IS	43
Appendix B. General Issues with Area Abstraction	44
Authors' Addresses	45

1. Introduction

This document describes a solution for IP/LDP fast-reroute [RFC5714]. MRT-FRR creates two alternate forwarding trees which are distinct from the primary next-hop forwarding used during stable operation. These two trees are maximally diverse from each other, providing link and node protection for 100% of paths and failures as long as the failure does not cut the network into multiple pieces. This document defines the architecture for IP/LDP fast-reroute with MRT.

[I-D.ietf-rtgwg-mrt-frr-algorithm] describes how to compute maximally redundant trees using a specific algorithm, the MRT Lowpoint algorithm. The MRT Lowpoint algorithm is used by a router that supports the Default MRT Profile, as specified in this document.

IP/LDP Fast-Reroute with MRT (MRT-FRR) uses two maximally diverse forwarding topologies to provide alternates. A primary next-hop should be on only one of the diverse forwarding topologies; thus, the other can be used to provide an alternate. Once traffic has been moved to one of the MRTs by one point of local repair (PLR), that traffic is not subject to further repair actions by another PLR, even in the event of multiple simultaneous failures. Therefore, traffic repaired by MRT-FRR will not loop between different PLRs responding to different simultaneous failures.

While MRT provides 100% protection for a single link or node failure, it may not protect traffic in the event of multiple simultaneous failures, nor does take into account Shared Risk Link Groups (SRLGs). Also, while the MRT Lowpoint algorithm is computationally efficient, it is also new. In order for MRT-FRR to function properly, all of the other nodes in the network that support MRT must correctly compute next-hops based on the same algorithm, and install the corresponding forwarding state. This is in contrast to other FRR methods where the calculation of backup paths generally involves repeated application of the simpler and widely-deployed shortest path first (SPF) algorithm, and backup paths themselves re-use the forwarding state used for shortest path forwarding of normal traffic. Section 14 provides operational guidance related to verification of MRT forwarding paths.

In addition to supporting IP and LDP unicast fast-reroute, the diverse forwarding topologies and guarantee of 100% coverage permit fast-reroute technology to be applied to multicast traffic as described in [I-D.atlas-rtgwg-mrt-mc-arch]. However, the current document does not address the multicast applications of MRTs.

1.1. Importance of 100% Coverage

Fast-reroute is based upon the single failure assumption - that the time between single failures is long enough for a network to reconverge and start forwarding on the new shortest paths. That does not imply that the network will only experience one failure or change.

It is straightforward to analyze a particular network topology for coverage. However, a real network does not always have the same topology. For instance, maintenance events will take links or nodes out of use. Simply costing out a link can have a significant effect on what loop-free alternates (LFAs) are available. Similarly, after a single failure has happened, the topology is changed and its associated coverage. Finally, many networks have new routers or links added and removed; each of those changes can have an effect on the coverage for topology-sensitive methods such as LFA and Remote

LFA. If fast-reroute is important for the network services provided, then a method that guarantees 100% coverage is important to accommodate natural network topology changes.

When a network needs to use Ordered FIB[RFC6976] or Nearside Tunneling[RFC5715] as a micro-loop prevention mechanism [RFC5715], then the whole IGP area needs to have alternates available. This allows the micro-loop prevention mechanism, which requires slower network convergence, to take the necessary time without adversely impacting traffic. Without complete coverage, traffic to the unprotected destinations will be dropped for significantly longer than with current convergence - where routers individually converge as fast as possible. See Section 12.1 for more discussion of micro-loop prevention and MRTs.

1.2. Partial Deployment and Backwards Compatibility

MRT-FRR supports partial deployment. Routers advertise their ability to support MRT. Inside the MRT-capable connected group of routers (referred to as an MRT Island), the MRTs are computed. Alternates to destinations outside the MRT Island are computed and depend upon the existence of a loop-free neighbor of the MRT Island for that destination. MRT Islands are discussed in detail in Section 7, and partial deployment is discussed in more detail in Section 14.5.

2. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

3. Terminology

network graph: A graph that reflects the network topology where all links connect exactly two nodes and broadcast links have been transformed into the standard pseudo-node representation.

cut-link: A link whose removal partitions the network. A cut-link by definition must be connected between two cut-vertices. If there are multiple parallel links, then they are referred to as cut-links in this document if removing the set of parallel links would partition the network graph.

cut-vertex: A vertex whose removal partitions the network graph.

2-connected: A graph that has no cut-vertices. This is a graph that requires two nodes to be removed before the network is partitioned.

2-connected cluster: A maximal set of nodes that are 2-connected.

block: Either a 2-connected cluster, a cut-edge, or an isolated vertex.

Redundant Trees (RT): A pair of trees where the path from any node X to the root R along the first tree is node-disjoint with the path from the same node X to the root along the second tree. Redundant trees can always be computed in 2-connected graphs.

Maximally Redundant Trees (MRT): A pair of trees where the path from any node X to the root R along the first tree and the path from the same node X to the root along the second tree share the minimum number of nodes and the minimum number of links. Each such shared node is a cut-vertex. Any shared links are cut-links. In graphs that are not 2-connected, it is not possible to compute RTs. However, it is possible to compute MRTs. MRTs are maximally redundant in the sense that they are as redundant as possible given the constraints of the network graph.

Directed Acyclic Graph (DAG): A graph where all links are directed and there are no cycles in it.

Almost Directed Acyclic Graph (ADAG): A graph with one node designated as the root. The graph has the property that if all links incoming to the root were removed, then resulting graph would be a DAG.

Generalized ADAG (GADAG): A graph that is the combination of the ADAGs of all blocks.

MRT-Red: MRT-Red is used to describe one of the two MRTs; it is used to describe the associated forwarding topology and MPLS multi-topology identifier (MT-ID). Specifically, MRT-Red is the decreasing MRT where links in the GADAG are taken in the direction from a higher topologically ordered node to a lower one.

MRT-Blue: MRT-Blue is used to describe one of the two MRTs; it is used to describe the associated forwarding topology and MPLS MT-ID. Specifically, MRT-Blue is the increasing MRT where links in the GADAG are taken in the direction from a lower topologically ordered node to a higher one.

Rainbow MRT: It is useful to have an MPLS MT-ID that refers to the multiple MRT forwarding topologies and to the default forwarding topology. This is referred to as the Rainbow MRT MPLS MT-ID and is used by LDP to reduce signaling and permit the same label to always be advertised to all peers for the same (MT-ID, Prefix).

MRT Island: The set of routers that support a particular MRT profile and the links connecting them that support MRT.

Island Border Router (IBR): A router in the MRT Island that is connected to a router not in the MRT Island and both routers are in a common area or level.

Island Neighbor (IN): A router that is not in the MRT Island but is adjacent to an IBR and in the same area/level as the IBR.

named proxy-node: A proxy-node can represent a destination prefix that can be attached to the MRT Island via at least two routers. It is named if there is a way that traffic can be encapsulated to reach specifically that proxy node; this could be because there is an LDP FEC (Forwarding Equivalence Class) for the associated prefix or because MRT-Red and MRT-Blue IP addresses are advertised in an undefined fashion for that proxy-node.

4. Maximally Redundant Trees (MRT)

A pair of Maximally Redundant Trees is a pair of directed spanning trees that provides maximally disjoint paths towards their common root. Only links or nodes whose failure would partition the network (i.e. cut-links and cut-vertices) are shared between the trees. The MRT Lowpoint algorithm is given in [I-D.ietf-rtgwg-mrt-frr-algorithm]. This algorithm can be computed in $O(e + n \log n)$; it is less than three SPF's. This document describes how the MRTs can be used and not how to compute them.

MRT provides destination-based trees for each destination. Each router stores its normal primary next-hop(s) as well as MRT-Blue next-hop(s) and MRT-Red next-hop(s) toward each destination. The alternate will be selected between the MRT-Blue and MRT-Red.

The most important thing to understand about MRTs is that for each pair of destination-routed MRTs, there is a path from every node X to the destination D on the Blue MRT that is as disjoint as possible from the path on the Red MRT.

For example, in Figure 1, there is a network graph that is 2-connected in (a) and associated MRTs in (b) and (c). One can consider the paths from B to R; on the Blue MRT, the paths are B->F->D->E->R or B->C->D->E->R. On the Red MRT, the path is B->A->R. These are clearly link and node-disjoint. These MRTs are redundant trees because the paths are disjoint.

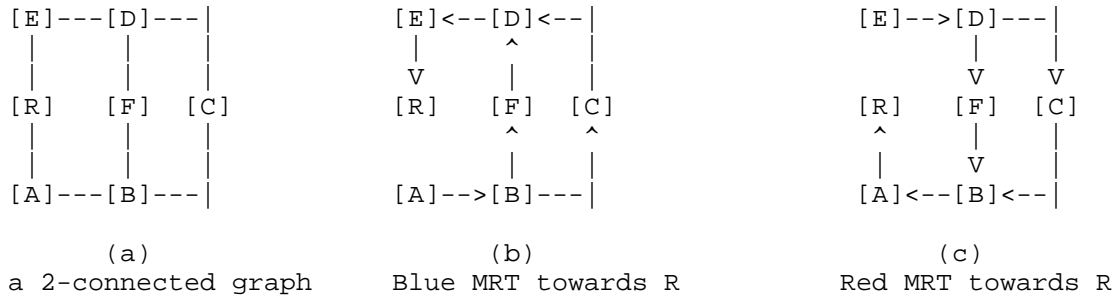


Figure 1: A 2-connected Network

By contrast, in Figure 2, the network in (a) is not 2-connected. If F, G or the link F<->G failed, then the network would be partitioned. It is clearly impossible to have two link-disjoint or node-disjoint paths from G, I or J to R. The MRTs given in (b) and (c) offer paths that are as disjoint as possible. For instance, the paths from B to R are the same as in Figure 1 and the path from G to R on the Blue MRT is G->F->D->E->R and on the Red MRT is G->F->B->A->R.

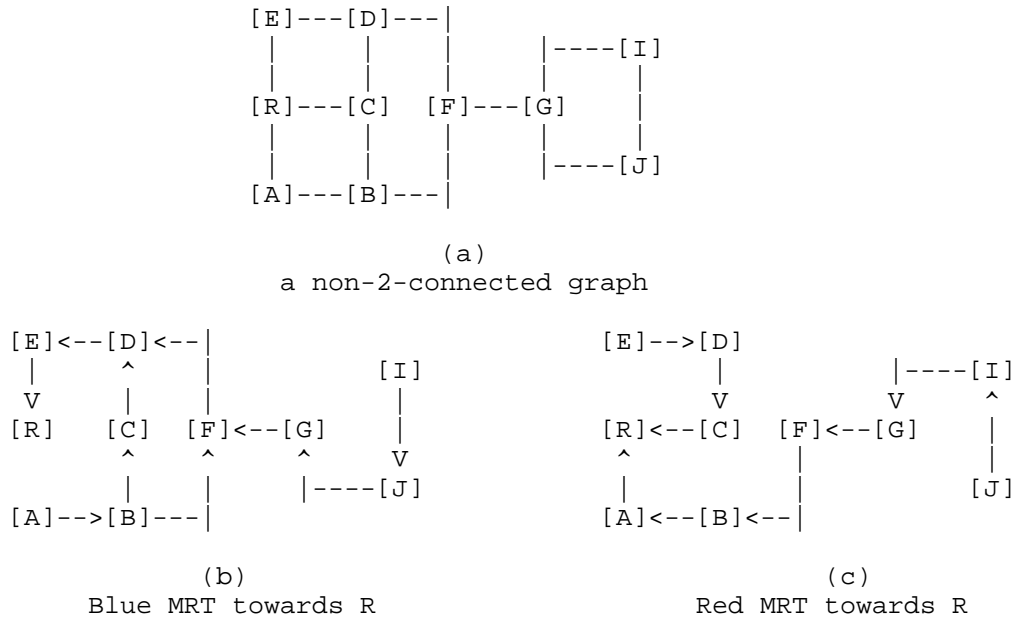


Figure 2: A non-2-connected network

5. Maximally Redundant Trees (MRT) and Fast-Reroute

In normal IGP routing, each router has its shortest path tree (SPT) to all destinations. From the perspective of a particular destination, D, this looks like a reverse SPT. To use maximally redundant trees, in addition, each destination D has two MRTs associated with it; by convention these will be called the MRT-Blue and MRT-Red. MRT-FRR is realized by using multi-topology forwarding. There is a MRT-Blue forwarding topology and a MRT-Red forwarding topology.

Any IP/LDP fast-reroute technique beyond LFA requires an additional dataplane procedure, such as an additional forwarding mechanism. The well-known options are multi-topology forwarding (used by MRT-FRR), tunneling (e.g. [RFC6981] or [RFC7490]), and per-interface forwarding (e.g. Loop-Free Failure Insensitive Routing in [EnyediThesis]).

When there is a link or node failure affecting, but not partitioning, the network, each node will still have at least one path via one of the MRTs to reach the destination D. For example, in Figure 2, C would normally forward traffic to R across the C->R link. If that C->R link fails, then C could use the Blue MRT path C->D->E->R.

As is always the case with fast-reroute technologies, forwarding does not change until a local failure is detected. Packets are forwarded along the shortest path. The appropriate alternate to use is pre-computed. [I-D.ietf-rtgwg-mrt-frr-algorithm] describes exactly how to determine whether the MRT-Blue next-hops or the MRT-Red next-hops should be the MRT alternate next-hops for a particular primary next-hop to a particular destination.

MRT alternates are always available to use. It is a local decision whether to use an MRT alternate, a Loop-Free Alternate or some other type of alternate.

As described in [RFC5286], when a worse failure than is anticipated happens, using LFAs that are not downstream neighbors can cause looping among alternates. Section 1.1 of [RFC5286] gives an example of link-protecting alternates causing a loop on node failure. Even if a worse failure than anticipated happens, the use of MRT alternates will not cause looping.

6. Unicast Forwarding with MRT Fast-Reroute

There are three possible types of routers involved in forwarding a packet along an MRT path. At the MRT ingress router, the packet leaves the shortest path to the destination and follows an MRT path

to the destination. In an FRR application, the MRT ingress router is the PLR. An MRT transit router takes a packet that arrives already associated with the particular MRT, and forwards it on that same MRT. In some situations (to be discussed later), the packet will need to leave the MRT path and return to the shortest path. This takes place at the MRT egress router. The MRT ingress and egress functionality may depend on the underlying type of packet being forwarded (LDP or IP). The MRT transit functionality is independent of the type of packet being forwarded. We first consider several MRT transit forwarding mechanisms. Then we look at how these forwarding mechanisms can be applied to carrying LDP and IP traffic.

6.1. Introduction to MRT Forwarding Options

The following options for MRT forwarding mechanisms are considered.

1. MRT LDP Labels

- A. Topology-scoped FEC encoded using a single label
- B. Topology and FEC encoded using a two label stack

2. MRT IP Tunnels

- A. MRT IPv4 Tunnels
- B. MRT IPv6 Tunnels

6.1.1. MRT LDP labels

We consider two options for the MRT forwarding mechanisms using MRT LDP labels.

6.1.1.1. Topology-scoped FEC encoded using a single label (Option 1A)

[RFC7307] provides a mechanism to distribute FEC-Label bindings scoped to a given MPLS topology (represented by MPLS MT-ID). To use multi-topology LDP to create MRT forwarding topologies, we associate two MPLS MT-IDs with the MRT-Red and MRT-Blue forwarding topologies, in addition to the default shortest path forwarding topology with MT-ID=0.

With this forwarding mechanism, a single label is distributed for each topology-scoped FEC. For a given FEC in the default topology (call it default-FEC-A), two additional topology-scoped FECs would be created, corresponding to the Red and Blue MRT forwarding topologies (call them red-FEC-A and blue-FEC-A). A router supporting this MRT transit forwarding mechanism advertises a different FEC-label binding

for each of the three topology-scoped FECs. When a packet is received with a label corresponding to red-FEC-A (for example), an MRT transit router will determine the next-hop for the MRT-Red forwarding topology for that FEC, swap the incoming label with the outgoing label corresponding to red-FEC-A learned from the MRT-Red next-hop router, and forward the packet.

This forwarding mechanism has the useful property that the FEC associated with the packet is maintained in the labels at each hop along the MRT. We will take advantage of this property when specifying how to carry LDP traffic on MRT paths using multi-topology LDP labels.

This approach is very simple for hardware to support. However, it reduces the label space for other uses, and it increases the memory needed to store the labels and the communication required by LDP to distribute FEC-label bindings. In general, this approach will also increase the time needed to install the FRR entries in the Forwarding Information Base (FIB) and hence the time needed before the next failure can be protected.

This forwarding option uses the LDP signaling extensions described in [RFC7307]. The MRT-specific LDP extensions required to support this option will be described elsewhere.

6.1.1.2. Topology and FEC encoded using a two label stack (Option 1B)

With this forwarding mechanism, a two label stack is used to encode the topology and the FEC of the packet. The top label (topology-id label) identifies the MRT forwarding topology, while the second label (FEC label) identifies the FEC. The top label would be a new FEC type with two values corresponding to MRT Red and Blue topologies.

When an MRT transit router receives a packet with a topology-id label, the router pops the top label and uses that it to guide the next-hop selection in combination with the next label in the stack (the FEC label). The router then swaps the FEC label, using the FEC-label bindings learned through normal LDP mechanisms. The router then pushes the topology-id label for the next-hop.

As with Option 1A, this forwarding mechanism also has the useful property that the FEC associated with the packet is maintained in the labels at each hop along the MRT.

This forwarding mechanism has minimal usage of additional labels, memory and LDP communication. It does increase the size of packets and the complexity of the required label operations and look-ups.

This forwarding option is consistent with context-specific label spaces, as described in [RFC5331]. However, the precise LDP behavior required to support this option for MRT has not been specified.

6.1.1.3. Compatibility of MRT LDP Label Options 1A and 1B

MRT transit forwarding based on MRT LDP Label options 1A and 1B can coexist in the same network, with a packet being forwarded along a single MRT path using the single label of option 1A for some hops and the two label stack of option 1B for other hops. However, to simplify the process of MRT Island formation we require that all routers in the MRT Island support at least one common forwarding mechanism. As an example, the Default MRT Profile requires support for the MRT LDP Label Option 1A forwarding mechanism. This ensures that the routers in an MRT island supporting the Default MRT Profile will be able to establish MRT forwarding paths based on MRT LDP Label Option 1A. However, an implementation supporting Option 1A may also support Option 1B. If the scaling or performance characteristics for the two options differ in this implementation, then it may be desirable for a pair of adjacent routers to use Option 1B labels instead of the Option 1A labels. If those routers successfully negotiate the use of Option 1B labels, they are free to use them. This can occur without any of the other routers in the MRT Island being made aware of it.

Note that this document only defines the Default MRT Profile which requires support for the MRT LDP Label Option 1A forwarding mechanism.

6.1.1.4. Required support for MRT LDP Label options

If a router supports a profile that includes the MRT LDP Label Option 1A for the MRT transit forwarding mechanism, then it **MUST** support option 1A, which encodes topology-scoped FECs using a single label. The router **MAY** also support option 1B.

If a router supports a profile that includes the MRT LDP Label Option 1B for the MRT transit forwarding mechanism, then it **MUST** support option 1B, which encodes the topology and FEC using a two label stack. The router **MAY** also support option 1A.

6.1.2. MRT IP tunnels (Options 2A and 2B)

IP tunneling can also be used as an MRT transit forwarding mechanism. Each router supporting this MRT transit forwarding mechanism announces two additional loopback addresses and their associated MRT color. Those addresses are used as destination addresses for MRT-blue and MRT-red IP tunnels respectively. The special loopback

addresses allow the transit nodes to identify the traffic as being forwarded along either the MRT-blue or MRT-red topology to reach the tunnel destination. For example, an MRT ingress router can cause a packet to be tunneled along the MRT-red path to router X by encapsulating the packet using the MRT-red loopback address advertised by router X. Upon receiving the packet, router X would remove the encapsulation header and forward the packet based on the original destination address.

Either IPv4 (option 2A) or IPv6 (option 2B) can be used as the tunneling mechanism.

Note that the two forwarding mechanisms using LDP Label options do not require additional loopbacks per router, as is required by the IP tunneling mechanism. This is because LDP labels are used on a hop-by-hop basis to identify MRT-blue and MRT-red forwarding topologies.

6.2. Forwarding LDP Unicast Traffic over MRT Paths

In the previous section, we examined several options for providing MRT transit forwarding functionality, which is independent of the type of traffic being carried. We now look at the MRT ingress functionality, which will depend on the type of traffic being carried (IP or LDP). We start by considering LDP traffic.

We also simplify the initial discussion by assuming that the network consists of a single IGP area, and that all routers in the network participate in MRT. Other deployment scenarios that require MRT egress functionality are considered later in this document.

In principle, it is possible to carry LDP traffic in MRT IP tunnels. However, for LDP traffic, it is desirable to avoid tunneling. Tunneling LDP traffic to a remote node requires knowledge of remote FEC-label bindings so that the LDP traffic can continue to be forwarded properly when it leaves the tunnel. This requires targeted LDP sessions which can add management complexity. As described below, the two MRT forwarding mechanisms that use LDP labels do not require targeted LDP sessions.

6.2.1. Forwarding LDP traffic using MRT LDP Label Option 1A

The MRT LDP Label option 1A forwarding mechanism uses topology-scoped FECs encoded using a single label as described in section Section 6.1.1.1. When a PLR receives an LDP packet that needs to be forwarded on the Red MRT (for example), it does a label swap operation, replacing the usual LDP label for the FEC with the Red MRT label for that FEC received from the next-hop router in the Red MRT computed by the PLR. When the next-hop router in the Red MRT

receives the packet with the Red MRT label for the FEC, the MRT transit forwarding functionality continues as described in Section 6.1.1.1. In this way the original FEC associated with the packet is maintained at each hop along the MRT.

6.2.2. Forwarding LDP traffic using MRT LDP Label Option 1B

The MRT LDP Label option 1B forwarding mechanism encodes the topology and the FEC using a two label stack as described in Section 6.1.1.2. When a PLR receives an LDP packet that needs to be forwarded on the Red MRT, it first does a normal LDP label swap operation, replacing the incoming normal LDP label associated with a given FEC with the outgoing normal LDP label for that FEC learned from the next-hop on the Red MRT. In addition, the PLR pushes the topology-identification label associated with the Red MRT, and forward the packet to the appropriate next-hop on the Red MRT. When the next-hop router in the Red MRT receives the packet with the Red MRT label for the FEC, the MRT transit forwarding functionality continues as described in Section 6.1.1.2. As with option 1A, the original FEC associated with the packet is maintained at each hop along the MRT.

6.2.3. Other considerations for forwarding LDP traffic using MRT LDP Labels

Note that forwarding LDP traffic using MRT LDP Labels can be done without the use of targeted LDP sessions when an MRT path to the destination FEC is used. The alternates selected in [I-D.ietf-rtgwg-mrt-frr-algorithm] use the MRT path to the destination FEC, so targeted LDP sessions are not needed. If instead one found it desirable to have the PLR use an MRT to reach the primary next-next-hop for the FEC, and then continue forwarding the LDP packet along the shortest path tree from the primary next-next-hop, this would require tunneling to the primary next-next-hop and a targeted LDP session for the PLR to learn the FEC-label binding for primary next-next-hop to correctly forward the packet.

6.2.4. Required support for LDP traffic

For greatest hardware compatibility, routers implementing MRT fast-reroute of LDP traffic MUST support Option 1A of encoding the MT-ID in the labels (See Section 9).

6.3. Forwarding IP Unicast Traffic over MRT Paths

For IPv4 traffic, there is no currently practical alternative except tunneling to gain the bits needed to indicate the MRT-Blue or MRT-Red forwarding topology. For IPv6 traffic, in principle one could define bits in the IPv6 options header to indicate the MRT-Blue or MRT-Red

forwarding topology. However, in this document, we have chosen not to define a solution that would work for IPv6 traffic but not for IPv4 traffic.

The choice of tunnel egress is flexible since any router closer to the destination than the next-hop can work. This architecture assumes that the original destination in the area is selected (see Section 11 for handling of multi-homed prefixes); another possible choice is the next-next-hop towards the destination. As discussed in the previous section, for LDP traffic, using the MRT to the original destination simplifies MRT-FRR by avoiding the need for targeted LDP sessions to the next-next-hop. For IP, that consideration doesn't apply.

Some situations require tunneling IP traffic along an MRT to a tunnel endpoint that is not the destination of the IP traffic. These situations will be discussed in detail later. We note here that an IP packet with a destination in a different IGP area/level from the PLR should be tunneled on the MRT to the Area Border Router (ABR) or Level Border Router (LBR) on the shortest path to the destination. For a destination outside of the PLR's MRT Island, the packet should be tunneled on the MRT to a non-proxy-node immediately before the named proxy-node on that particular color MRT.

6.3.1. Tunneling IP traffic using MRT LDP Labels

An IP packet can be tunneled along an MRT path by pushing the appropriate MRT LDP label(s). Tunneling using LDP labels, as opposed to IP headers, has the advantage that more installed routers can do line-rate encapsulation and decapsulation using LDP than using IP. Also, no additional IP addresses would need to be allocated or signaled.

6.3.1.1. Tunneling IP traffic using MRT LDP Label Option 1A

The MRT LDP Label option 1A forwarding mechanism uses topology-scoped FECs encoded using a single label as described in section Section 6.1.1.1. When a PLR receives an IP packet that needs to be forwarded on the Red MRT to a particular tunnel endpoint, it does a label push operation. The label pushed is the Red MRT label for a FEC originated by the tunnel endpoint, learned from the next-hop on the Red MRT.

6.3.1.2. Tunneling IP traffic using MRT LDP Label Option 1B

The MRT LDP Label option 1B forwarding mechanism encodes the topology and the FEC using a two label stack as described in Section 6.1.1.2. When a PLR receives an IP packet that needs to be forwarded on the

Red MRT to a particular tunnel endpoint, the PLR pushes two labels on the IP packet. The first (inner) label is the normal LDP label learned from the next-hop on the Red MRT, associated with a FEC originated by the tunnel endpoint. The second (outer) label is the topology-identification label associated with the Red MRT.

For completeness, we note here a potential variation that uses a single label as opposed to two labels. In order to tunnel an IP packet over an MRT to the destination of the IP packet (as opposed to an arbitrary tunnel endpoint), then we could just push a topology-identification label directly onto the packet. An MRT transit router would need to pop the topology-id label, do an IP route lookup in the context of that topology-id, and push the topology-id label.

6.3.2. Tunneling IP traffic using MRT IP Tunnels

In order to tunnel over the MRT to a particular tunnel endpoint, the PLR encapsulates the original IP packet with an additional IP header using the MRT-Blue or MRT-Red loopback address of the tunnel endpoint.

6.3.3. Required support for IP traffic

For greatest hardware compatibility and ease in removing the MRT-topology marking at area/level boundaries, routers that support MPLS and implement IP MRT fast-reroute MUST support tunneling of IP traffic using MRT LDP Label Option 1A (topology-scoped FEC encoded using a single label).

7. MRT Island Formation

The purpose of communicating support for MRT is to indicate that the MRT-Blue and MRT-Red forwarding topologies are created for transit traffic. The MRT architecture allows for different, potentially incompatible options. In order to create consistent MRT forwarding topologies, the routers participating in a particular MRT Island need to use the same set of options. These options are grouped into MRT profiles. In addition, the routers in an MRT Island all need to use the same set of nodes and links within the Island when computing the MRT forwarding topologies. This section describes the information used by a router to determine the nodes and links to include in a particular MRT Island. Some information already exists in the IGP and can be used by MRT in Island formation, subject to the interpretation defined here.

Other information needs to be communicated between routers for which there do not currently exist protocol extensions. This new information needs to be shared among all routers in an IGP area, so

defining extensions to existing IGPs to carry this information makes sense. These new protocol extensions will be defined elsewhere.

Deployment scenarios using multi-topology OSPF or IS-IS, or running both IS-IS and OSPF on the same routers is out of scope for this specification. As with LFA, it is expected that OSPF Virtual Links will not be supported.

At a high level, an MRT Island is defined as the set of routers supporting the same MRT profile, in the same IGP area/level and the bi-directional links interconnecting those routers. More detailed descriptions of these criteria are given below.

7.1. IGP Area or Level

All links in an MRT Island are bidirectional and belong to the same IGP area or level. For IS-IS, a link belonging to both level 1 and level 2 would qualify to be in multiple MRT Islands. A given ABR or LBR can belong to multiple MRT Islands, corresponding to the areas or levels in which it participates. Inter-area forwarding behavior is discussed in Section 10.

7.2. Support for a specific MRT profile

All routers in an MRT Island support the same MRT profile. A router advertises support for a given MRT profile using an 8-bit MRT Profile ID value. The registry for the MRT Profile ID is defined in this document. The protocol extensions for advertising the MRT Profile ID value will be defined elsewhere. A given router can support multiple MRT profiles and participate in multiple MRT Islands. The options that make up an MRT profile, as well as the default MRT profile, are defined in Section 8.

The process of MRT Island formation takes place independently for each MRT profile advertised by a given router. For example, consider a network with 40 connected routers in the same area advertising support for MRT Profile A and MRT Profile B. Two distinct MRT Islands will be formed corresponding to Profile A and Profile B, with each island containing all 40 routers. A complete set of maximally redundant trees will be computed for each island following the rules defined for each profile. If we add a third MRT Profile to this example, with Profile C being advertised by a connected subset of 30 routers, there will be a third MRT Island formed corresponding to those 30 routers, and a third set of maximally redundant trees will be computed. In this example, 40 routers would compute and install two sets of MRT transit forwarding entries corresponding to Profiles A and B, while 30 routers would compute and install three sets of MRT transit forwarding entries corresponding to Profiles A, B, and C.

7.3. Excluding additional routers and interfaces from the MRT Island

MRT takes into account existing IGP mechanisms for discouraging traffic from using particular links and routers, and it introduces an MRT-specific exclusion mechanism for links.

7.3.1. Existing IGP exclusion mechanisms

Mechanisms for discouraging traffic from using particular links already exist in IS-IS and OSPF. In IS-IS, an interface configured with a metric of $2^{24}-2$ (0xFFFFFE) will only be used as a last resort. (An interface configured with a metric of $2^{24}-1$ (0xFFFFF) will not be advertised into the topology.) In OSPF, an interface configured with a metric of $2^{16}-1$ (0xFFFF) will only be used as a last resort. These metrics can be configured manually to enforce administrative policy, or they can be set in an automated manner as with LDP IGP synchronization [RFC5443].

Mechanisms also already exist in IS-IS and OSPF to discourage or prevent transit traffic from using a particular router. In IS-IS, the overload bit is prevents transit traffic from using a router.

For OSPFv2 and OSPFv3, [RFC6987] specifies setting all outgoing interface metrics to 0xFFFF to discourage transit traffic from using a router. ([RFC6987] defines the metric value 0xFFFF as MaxLinkMetric, a fixed architectural value for OSPF.) For OSPFv3, [RFC5340] specifies that a router be excluded from the intra-area shortest path tree computation if the V6-bit or R-bit of the LSA options is not set in the Router LSA.

The following rules for MRT Island formation ensure that MRT FRR protection traffic does not use a link or router that is discouraged or prevented from carrying traffic by existing IGP mechanisms.

1. A bidirectional link MUST be excluded from an MRT Island if either the forward or reverse cost on the link is 0xFFFFFE (for IS-IS) or 0xFFFF for OSPF.
2. A router MUST be excluded from an MRT Island if it is advertised with the overload bit set (for IS-IS), or it is advertised with metric values of 0xFFFF on all of its outgoing interfaces (for OSPFv2 and OSPFv3).
3. A router MUST be excluded from an MRT Island if it is advertised with either the V6-bit or R-bit of the LSA options not set in the Router LSA.

7.3.2. MRT-specific exclusion mechanism

This architecture also defines a means of excluding an otherwise usable link from MRT Islands. The protocol extensions for advertising that a link is MRT-Ineligible will be defined elsewhere. A link with either interface advertised as MRT-Ineligible MUST be excluded from an MRT Island. Note that an interface advertised as MRT-Ineligible by a router is ineligible with respect to all profiles advertised by that router.

7.4. Connectivity

All of the routers in an MRT Island MUST be connected by bidirectional links with other routers in the MRT Island. Disconnected MRT Islands will operate independently of one another.

7.5. Algorithm for MRT Island Identification

An algorithm that allows a computing router to identify the routers and links in the local MRT Island satisfying the above rules is given in section 5.2 of [I-D.ietf-rtgwg-mrt-frr-algorithm].

8. MRT Profile

An MRT Profile is a set of values and options related to MRT behavior. The complete set of options is designated by the corresponding 8-bit Profile ID value.

This document specifies the values and options that correspond to the Default MRT Profile (Profile ID = 0). Future documents may define other MRT Profiles by specifying the MRT Profile Options below.

8.1. MRT Profile Options

Below is a description of the values and options that define an MRT Profile.

MRT Algorithm: This identifies the particular algorithm for computing maximally redundant trees used by the router for this profile.

MRT-Red MT-ID: This specifies the MPLS MT-ID to be associated with the MRT-Red forwarding topology. It is allocated from the MPLS Multi-Topology Identifiers Registry.

MRT-Blue MT-ID: This specifies the MPLS MT-ID to be associated with the MRT-Blue forwarding topology. It is allocated from the MPLS Multi-Topology Identifiers Registry.

GADAG Root Selection Policy: This specifies the manner in which the GADAG root is selected. All routers in the MRT island need to use the same GADAG root in the calculations used to construct the MRTs. A valid GADAG Root Selection Policy **MUST** be such that each router in the MRT island chooses the same GADAG root based on information available to all routers in the MRT island. GADAG Root Selection Priority values, advertised as router-specific MRT parameters, **MAY** be used in a GADAG Root Selection Policy.

MRT Forwarding Mechanism: This specifies which forwarding mechanism the router uses to carry transit traffic along MRT paths. A router which supports a specific MRT forwarding mechanism must program appropriate next-hops into the forwarding plane. The current options are MRT LDP Label Option 1A, MRT LDP Label Option 1B, IPv4 Tunneling, IPv6 Tunneling, and None. If IPv4 is supported, then both MRT-Red and MRT-Blue IPv4 Loopback Addresses **SHOULD** be specified. If IPv6 is supported, both MRT-Red and MRT-Blue IPv6 Loopback Addresses **SHOULD** be specified.

Recalculation: Recalculation specifies the process and timing by which new MRTs are computed after the topology has been modified.

Area/Level Border Behavior: This specifies how traffic traveling on the MRT-Blue or MRT-Red in one area should be treated when it passes into another area.

Other Profile-Specific Behavior: Depending upon the use-case for the profile, there may be additional profile-specific behavior.

When a new MRT Profile is defined, new and unique values should be allocated from the MPLS Multi-Topology Identifiers Registry, corresponding to the MRT-Red and MRT-Blue MT-ID values for the new MRT Profile .

If a router advertises support for multiple MRT profiles, then it **MUST** create the transit forwarding topologies for each of those, unless the profile specifies the None option for MRT Forwarding Mechanism.

The ability of MRT-FRR to support transit forwarding entries for multiple profiles can be used to facilitate a smooth transition from an existing deployed MRT Profile to a new MRT Profile. The new profile can be activated in parallel with the existing profile, installing the transit forwarding entries for the new profile without affecting the transit forwarding entries for the existing profile. Once the new transit forwarding state has been verified, the router can be configured to use the alternates computed by the new profile in the event of a failure.

8.2. Router-specific MRT parameters

For some profiles, additional router-specific MRT parameters may need to be advertised. While the set of options indicated by the MRT Profile ID must be identical for all routers in an MRT Island, these router-specific MRT parameters may differ between routers in the same MRT island. Several such parameters are described below.

GADAG Root Selection Priority: A GADAG Root Selection Policy MAY rely on the GADAG Root Selection Priority values advertised by each router in the MRT island. A GADAG Root Selection Policy may use the GADAG Root Selection Priority to allow network operators to configure a parameter to ensure that the GADAG root is selected from a particular subset of routers. An example of this use of the GADAG Root Selection Priority value by the GADAG Root Selection Policy is given in the Default MRT profile below.

MRT-Red Loopback Address: This provides the router's loopback address to reach the router via the MRT-Red forwarding topology. It can be specified for either IPv4 or IPv6. Note that this parameter is not needed to support the Default MRT profile.

MRT-Blue Loopback Address: This provides the router's loopback address to reach the router via the MRT-Blue forwarding topology. It can be specified for either IPv4 and IPv6. Note that this parameter is not needed to support the Default MRT profile.

Protocol extensions for advertising a router's GADAG Root Selection Priority value will be defined in other documents. Protocol extensions for the advertising a router's MRT-Red and MRT-Blue Loopback Addresses will be defined elsewhere.

8.3. Default MRT profile

The following set of options defines the default MRT Profile. The default MRT profile is indicated by the MRT Profile ID value of 0.

MRT Algorithm: MRT Lowpoint algorithm defined in [I-D.ietf-rtgwg-mrt-frr-algorithm].

MRT-Red MPLS MT-ID: This value will be allocated from the MPLS Multi-Topology Identifiers Registry. The IANA request for this allocation will be in another document.

MRT-Blue MPLS MT-ID: This value will be allocated from the MPLS Multi-Topology Identifiers Registry. The IANA request for this allocation will be in another document.

GADAG Root Selection Policy: Among the routers in the MRT Island with the lowest numerical value advertised for GADAG Root Selection Priority, an implementation MUST pick the router with the highest Router ID to be the GADAG root. Note that a lower numerical value for GADAG Root Selection Priority indicates a higher preference for selection.

Forwarding Mechanisms: MRT LDP Label Option 1A

Recalculation: Recalculation of MRTs SHOULD occur as described in Section 12.2. This allows the MRT forwarding topologies to support IP/LDP fast-reroute traffic.

Area/Level Border Behavior: As described in Section 10, ABRs/LBRs SHOULD ensure that traffic leaving the area also exits the MRT-Red or MRT-Blue forwarding topology.

9. LDP signaling extensions and considerations

The protocol extensions for LDP will be defined in another document. A router must indicate that it has the ability to support MRT; having this explicit allows the use of MRT-specific processing, such as special handling of FECs sent with the Rainbow MRT MT-ID.

A FEC sent with the Rainbow MRT MT-ID indicates that the FEC applies to all the MRT-Blue and MRT-Red MT-IDs in supported MRT profiles. The FEC-label bindings for the default shortest-path based MT-ID 0 MUST still be sent (even though it could be inferred from the Rainbow FEC-label bindings) to ensure continuous operation of normal LDP forwarding. The Rainbow MRT MT-ID is defined to provide an easy way to handle the special signaling that is needed at ABRs or LBRs. It avoids the problem of needing to signal different MPLS labels to different LDP neighbors for the same FEC. Because the Rainbow MRT MT-ID is used only by ABRs/LBRs or an LDP egress router, it is not MRT profile specific.

The value of the Rainbow MRT MPLS MT-ID will be allocated from the MPLS Multi-Topology Identifiers Registry. The IANA request for this allocation will be in another document.

10. Inter-area Forwarding Behavior

An ABR/LBR has two forwarding roles. First, it forwards traffic within areas. Second, it forwards traffic from one area into another. These same two roles apply for MRT transit traffic. Traffic on MRT-Red or MRT-Blue destined inside the area needs to stay on MRT-Red or MRT-Blue in that area. However, it is desirable for

traffic leaving the area to also exit MRT-Red or MRT-Blue and return to shortest path forwarding.

For unicast MRT-FRR, the need to stay on an MRT forwarding topology terminates at the ABR/LBR whose best route is via a different area/level. It is highly desirable to go back to the default forwarding topology when leaving an area/level. There are three basic reasons for this. First, the default topology uses shortest paths; the packet will thus take the shortest possible route to the destination. Second, this allows a single router failure that manifests itself in multiple areas (as would be the case with an ABR/LBR failure) to be separately identified and repaired around. Third, the packet can be fast-rerouted again, if necessary, due to a second distinct failure in a different area.

In OSPF, an ABR that receives a packet on MRT-Red or MRT-Blue towards destination Z should continue to forward the packet along MRT-Red or MRT-Blue only if the best route to Z is in the same OSPF area as the interface that the packet was received on. Otherwise, the packet should be removed from MRT-Red or MRT-Blue and forwarded on the shortest-path default forwarding topology.

The above description applies to OSPF. The same essential behavior also applies to IS-IS if one substitutes IS-IS level for OSPF area. However, the analogy with OSPF is not exact. An interface in OSPF can only be in one area, whereas an interface in IS-IS can be in both Level-1 and Level-2. Therefore, to avoid confusion and address this difference, we explicitly describe the behavior for IS-IS in Appendix A. In the following sections only the OSPF terminology is used.

10.1. ABR Forwarding Behavior with MRT LDP Label Option 1A

For LDP forwarding where a single label specifies (MT-ID, FEC), the ABR is responsible for advertising the proper label to each neighbor. Assume that an ABR has allocated three labels for a particular destination; those labels are *L_primary*, *L_blue*, and *L_red*. To those routers in the same area as the best route to the destination, the ABR advertises the following FEC-label bindings: *L_primary* for the default topology, *L_blue* for the MRT-Blue MT-ID and *L_red* for the MRT-Red MT-ID, as expected. However, to routers in other areas, the ABR advertises the following FEC-label bindings: *L_primary* for the default topology, and *L_primary* for the Rainbow MRT MT-ID. Associating *L_primary* with the Rainbow MRT MT-ID causes the receiving routers to use *L_primary* for the MRT-Blue MT-ID and for the MRT-Red MT-ID.

The ABR installs all next-hops for the best area: primary next-hops for `L_primary`, MRT-Blue next-hops for `L_blue`, and MRT-Red next-hops for `L_red`. Because the ABR advertised (Rainbow MRT MT-ID, FEC) with `L_primary` to neighbors not in the best area, packets from those neighbors will arrive at the ABR with a label `L_primary` and will be forwarded into the best area along the default topology. By controlling what labels are advertised, the ABR can thus enforce that packets exiting the area do so on the shortest-path default topology.

10.1.1. Motivation for Creating the Rainbow-FEC

The desired forwarding behavior could be achieved in the above example without using the Rainbow-FEC. This could be done by having the ABR advertise the following FEC-label bindings to neighbors not in the best area: `L1_primary` for the default topology, `L1_primary` for the MRT-Blue MT-ID, and `L1_primary` for the MRT-Red MT-ID. Doing this would require machinery to spoof the labels used in FEC-label binding advertisements on a per-neighbor basis. Such label-spoofing machinery does not currently exist in most LDP implementations and doesn't have other obvious uses.

Many existing LDP implementations do however have the ability to filter FEC-label binding advertisements on a per-neighbor basis. The Rainbow-FEC allows us to re-use the existing per-neighbor FEC filtering machinery to achieve the desired result. By introducing the Rainbow FEC, we can use per-neighbor FEC-filtering machinery to advertise the FEC-label binding for the Rainbow-FEC (and filter those for MRT-Blue and MRT-Red) to non-best-area neighbors of the ABR.

An ABR may choose to either advertise the Rainbow-FEC or advertise separate MRT-Blue and MRT-Red advertisements. This is a local choice. A router that supports the MRT LDP Label Option 1A Forwarding Mechanism MUST be able to receive and correctly interpret the Rainbow-FEC.

10.2. ABR Forwarding Behavior with IP Tunneling (option 2)

If IP tunneling is used, then the ABR behavior is dependent upon the outermost IP address. If the outermost IP address is an MRT loopback address of the ABR, then the packet is decapsulated and forwarded based upon the inner IP address, which should go on the default SPT topology. If the outermost IP address is not an MRT loopback address of the ABR, then the packet is simply forwarded along the associated forwarding topology. A PLR sending traffic to a destination outside its local area/level will pick the MRT and use the associated MRT loopback address of the selected ABR advertising the lowest cost to the external destination.

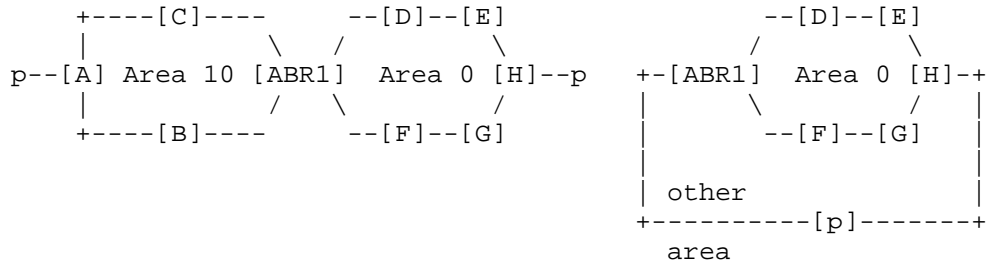
Thus, for these two MRT Forwarding Mechanisms (MRT LDP Label option 1A and IP tunneling option 2), there is no need for additional computation or per-area forwarding state.

10.3. ABR Forwarding Behavior with MRT LDP Label option 1B

The other MRT forwarding mechanism described in Section 6 uses two labels, a topology-id label, and a FEC-label. This mechanism would require that any router whose MRT-Red or MRT-Blue next-hop is an ABR would need to determine whether the ABR would forward the packet out of the area/level. If so, then that router should pop off the topology-identification label before forwarding the packet to the ABR.

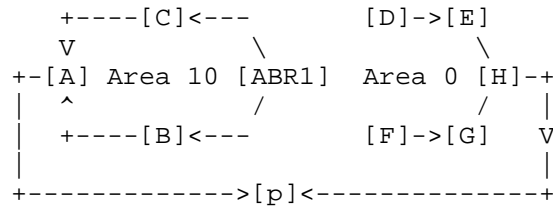
For example, in Figure 3, if node H fails, node E has to put traffic towards prefix p onto MRT-Red. But since node D knows that ABR1 will use a best route from another area, it is safe for D to pop the Topology-Identification Label and just forward the packet to ABR1 along the MRT-Red next-hop. ABR1 will use the shortest path in Area 10.

In all cases for IS-IS and most cases for OSPF, the penultimate router can determine what decision the adjacent ABR will make. The one case where it can't be determined is when two ASBRs are in different non-backbone areas attached to the same ABR, then the ASBR's Area ID may be needed for tie-breaking (prefer the route with the largest OPSF area ID) and the Area ID isn't announced as part of the ASBR link-state advertisement (LSA). In this one case, suboptimal forwarding along the MRT in the other area would happen. If that becomes a realistic deployment scenario, protocol extensions could be developed to address this issue.

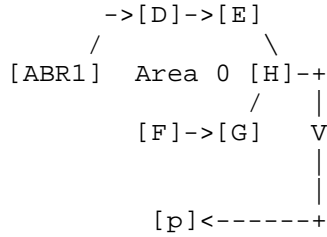


(a) Example topology

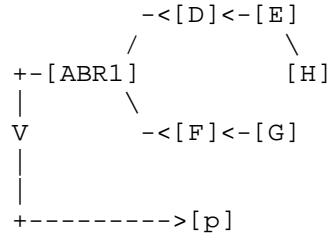
(b) Proxy node view in Area 0 nodes



(c) rSPT towards destination p



(d) Blue MRT in Area 0



(e) Red MRT in Area 0

Figure 3: ABR Forwarding Behavior and MRTs

11. Prefixes Multiply Attached to the MRT Island

How a computing router S determines its local MRT Island for each supported MRT profile is already discussed in Section 7.

There are two types of prefixes or FECs that may be multiply attached to an MRT Island. The first type are multi-homed prefixes that usually connect at a domain or protocol boundary. The second type represent routers that do not support the profile for the MRT Island.

The key difference is whether the traffic, once out of the MRT Island, might re-enter the MRT Island if a loop-free exit point is not selected.

FRR using LFA has the useful property that it is able to protect multi-homed prefixes against ABR failure. For instance, if a prefix from the backbone is available via both ABR A and ABR B, if A fails, then the traffic should be redirected to B. This can be accomplished with MRT FRR as well.

If ASBR protection is desired, this has additional complexities if the ASBRs are in different areas. Similarly, protecting labeled BGP traffic in the event of an ASBR failure has additional complexities due to the per-ASBR label spaces involved.

As discussed in [RFC5286], a multi-homed prefix could be:

- o An out-of-area prefix announced by more than one ABR,
- o An AS-External route announced by 2 or more ASBRs,
- o A prefix with iBGP multipath to different ASBRs,
- o etc.

See Appendix B for a discussion of a general issue with multi-homed prefixes connected in two different areas.

There are also two different approaches to protection. The first is tunnel endpoint selection where the PLR picks a router to tunnel to where that router is loop-free with respect to the failure-point. Conceptually, the set of candidate routers to provide LFAs expands to all routers that can be reached via an MRT alternate, attached to the prefix.

The second is to use a proxy-node, that can be named via MPLS label or IP address, and pick the appropriate label or IP address to reach it on either MRT-Blue or MRT-Red as appropriate to avoid the failure point. A proxy-node can represent a destination prefix that can be attached to the MRT Island via at least two routers. It is termed a named proxy-node if there is a way that traffic can be encapsulated to reach specifically that proxy-node; this could be because there is an LDP FEC for the associated prefix or because MRT-Red and MRT-Blue IP addresses are advertised (in an as-yet undefined fashion) for that proxy-node. Traffic to a named proxy-node may take a different path than traffic to the attaching router; traffic is also explicitly forwarded from the attaching router along a predetermined interface towards the relevant prefixes.

For IP traffic, multi-homed prefixes can use tunnel endpoint selection. For IP traffic that is destined to a router outside the MRT Island, if that router is the egress for a FEC advertised into the MRT Island, then the named proxy-node approach can be used.

For LDP traffic, there is always a FEC advertised into the MRT Island. The named proxy-node approach should be used, unless the computing router S knows the label for the FEC at the selected tunnel endpoint.

If a FEC is advertised from outside the MRT Island into the MRT Island and the forwarding mechanism specified in the profile includes LDP, then the routers learning that FEC MUST also advertise labels for (MRT-Red, FEC) and (MRT-Blue, FEC) to neighbors inside the MRT Island. Any router receiving a FEC corresponding to a router outside the MRT Island or to a multi-homed prefix MUST compute and install the transit MRT-Blue and MRT-Red next-hops for that FEC. The FEC-label bindings for the topology-scoped FECs ((MT-ID 0, FEC), (MRT-Red, FEC), and (MRT-Blue, FEC)) MUST also be provided via LDP to neighbors inside the MRT Island.

11.1. Protecting Multi-Homed Prefixes using Tunnel Endpoint Selection

Tunnel endpoint selection is a local matter for a router in the MRT Island since it pertains to selecting and using an alternate and does not affect the transit MRT-Red and MRT-Blue forwarding topologies.

Let the computing router be S and the next-hop F be the node whose failure is to be avoided. Let the destination be prefix p. Have A be the router to which the prefix p is attached for S's shortest path to p.

The candidates for tunnel endpoint selection are those to which the destination prefix is attached in the area/level. For a particular candidate B, it is necessary to determine if B is loop-free to reach p with respect to S and F for node-protection or at least with respect to S and the link (S, F) for link-protection. If B will always prefer to send traffic to p via a different area/level, then this is definitional. Otherwise, distance-based computations are necessary and an SPF from B's perspective may be necessary. The following equations give the checks needed; the rationale is similar to that given in [RFC5286]. In the inequalities below, $D_{opt}(X,Y)$ means the shortest distance from node X to node Y, and $D_{opt}(X,p)$ means the shortest distance from node X to prefix p.

Loop-Free for S: $D_{opt}(B, p) < D_{opt}(B, S) + D_{opt}(S, p)$

Loop-Free for F: $D_{opt}(B, p) < D_{opt}(B, F) + D_{opt}(F, p)$

The latter is equivalent to the following, which avoids the need to compute the shortest path from F to p.

Loop-Free for F: $D_{\text{opt}}(B, p) < D_{\text{opt}}(B, F) + D_{\text{opt}}(S, p) - D_{\text{opt}}(S, F)$

Finally, the rules for Endpoint selection are given below. The basic idea is to repair to the prefix-advertising router selected for the shortest-path and only to select and tunnel to a different endpoint if necessary (e.g. $A=F$ or F is a cut-vertex or the link (S,F) is a cut-link).

1. Does S have a node-protecting alternate to A? If so, select that. Tunnel the packet to A along that alternate. For example, if LDP is the forwarding mechanism, then push the label (MRT-Red, A) or (MRT-Blue, A) onto the packet.
2. If not, then is there a router B that is loop-free to reach p while avoiding both F and S? If so, select B as the end-point. Determine the MRT alternate to reach B while avoiding F. Tunnel the packet to B along that alternate. For example, with LDP, push the label (MRT-Red, B) or (MRT-Blue, B) onto the packet.
3. If not, then does S have a link-protecting alternate to A? If so, select that.
4. If not, then is there a router B that is loop-free to reach p while avoiding S and the link from S to F? If so, select B as the endpoint and the MRT alternate for reaching B from S that avoid the link (S,F).

The tunnel endpoint selected will receive a packet destined to itself and, being the egress, will pop that MPLS label (or have signaled Implicit Null) and forward based on what is underneath. This suffices for IP traffic since the tunnel endpoint can use the IP header of the original packet to continue forwarding the packet. However, tunnelling of LDP traffic requires targeted LDP sessions for learning the FEC-label binding at the tunnel endpoint.

11.2. Protecting Multi-Homed Prefixes using Named Proxy-Nodes

Instead, the named proxy-node method works with LDP traffic without the need for targeted LDP sessions. It also has a clear advantage over tunnel endpoint selection, in that it is possible to explicitly forward from the MRT Island along an interface to a loop-free island neighbor when that interface may not be a primary next-hop.

A named proxy-node represents one or more destinations and, for LDP forwarding, has a FEC associated with it that is signalled into the MRT Island. Therefore, it is possible to explicitly label packets to go to (MRT-Red, FEC) or (MRT-Blue, FEC); at the border of the MRT Island, the label will swap to meaning (MT-ID 0, FEC). It would be possible to have named proxy-nodes for IP forwarding, but this would require extensions to signal two IP addresses to be associated with MRT-Red and MRT-Blue for the proxy-node. A named proxy-node can be uniquely represented by the two routers in the MRT Island to which it is connected. The extensions to signal such IP addresses will be defined elsewhere. The details of what label-bindings must be originated will be described in another document.

Computing the MRT next-hops to a named proxy-node and the MRT alternate for the computing router S to avoid a particular failure node F is straightforward. The details of the simple constant-time functions, `Select_Proxy_Node_NHs()` and `Select_Alternates_Proxy_Node()`, are given in [I-D.ietf-rtgwg-mrt-frr-algorithm]. A key point is that computing these MRT next-hops and alternates can be done as new named proxy-nodes are added or removed without requiring a new MRT computation or impacting other existing MRT paths. This maps very well to, for example, how OSPFv2 (see [RFC2328] Section 16.5) does incremental updates for new summary-LSAs.

The remaining question is how to attach the named proxy-node to the MRT Island; all the routers in the MRT Island MUST do this consistently. No more than 2 routers in the MRT Island can be selected; one should only be selected if there are no others that meet the necessary criteria. The named proxy-node is logically part of the area/level.

There are two sources for candidate routers in the MRT Island to connect to the named proxy-node. The first set are those routers in the MRT Island that are advertising the prefix; the named-proxy-cost assigned to each prefix-advertising router is the announced cost to the prefix. The second set are those routers in the MRT Island that are connected to routers not in the MRT Island but in the same area/level; such routers will be defined as Island Border Routers (IBRs). The routers connected to the IBRs that are not in the MRT Island and are in the same area/level as the MRT island are Island Neighbors (INs).

Since packets sent to the named proxy-node along MRT-Red or MRT-Blue may come from any router inside the MRT Island, it is necessary that whatever router to which an IBR forwards the packet be loop-free with respect to the whole MRT Island for the destination. Thus, an IBR is a candidate router only if it possesses at least one IN whose

shortest path to the prefix does not enter the MRT Island. A method for identifying loop-free Island Neighbors(LFINs) is given in [I-D.ietf-rtgwg-mrt-frr-algorithm]. The named-proxy-cost assigned to each (IBR, IN) pair is $\text{cost}(\text{IBR}, \text{IN}) + D_{\text{opt}}(\text{IN}, \text{prefix})$.

From the set of prefix-advertising routers and the set of IBRs with at least one LFIN, the two routers with the lowest named-proxy-cost are selected. Ties are broken based upon the lowest Router ID. For ease of discussion, the two selected routers will be referred to as proxy-node attachment routers.

A proxy-node attachment router has a special forwarding role. When a packet is received destined to (MRT-Red, prefix) or (MRT-Blue, prefix), if the proxy-node attachment router is an IBR, it MUST swap to the shortest path forwarding topology (e.g. swap to the label for (MT-ID 0, prefix) or remove the outer IP encapsulation) and forward the packet to the IN whose cost was used in the selection. If the proxy-node attachment router is not an IBR, then the packet MUST be removed from the MRT forwarding topology and sent along the interface(s) that caused the router to advertise the prefix; this interface might be out of the area/level/AS.

11.3. MRT Alternates for Destinations Outside the MRT Island

A natural concern with new functionality is how to have it be useful when it is not deployed across an entire IGP area. In the case of MRT FRR, where it provides alternates when appropriate LFAs aren't available, there are also deployment scenarios where it may make sense to only enable some routers in an area with MRT FRR. A simple example of such a scenario would be a ring of 6 or more routers that is connected via two routers to the rest of the area.

Destinations inside the local island can obviously use MRT alternates. Destinations outside the local island can be treated like a multi-homed prefix and either Endpoint Selection or Named Proxy-Nodes can be used. Named Proxy-Nodes MUST be supported when LDP forwarding is supported and a label-binding for the destination is sent to an IBR.

Naturally, there are more complicated options to improve coverage, such as connecting multiple MRT islands across tunnels, but the need for the additional complexity has not been justified.

12. Network Convergence and Preparing for the Next Failure

After a failure, MRT detours ensure that packets reach their intended destination while the IGP has not reconverged onto the new topology. As link-state updates reach the routers, the IGP process calculates

the new shortest paths. Two things need attention: micro-loop prevention and MRT re-calculation.

12.1. Micro-loop prevention and MRTs

A micro-loop is a transient packet forwarding loop among two or more routers that can occur during convergence of IGP forwarding state. [RFC5715] discusses several techniques for preventing micro-loops. This section discusses how MRT-FRR relates to two of the micro-loop prevention techniques discussed in [RFC5715], Nearside Tunneling and Farside Tunneling.

In Nearside Tunneling, a router (PLR) adjacent to a failure perform local repair and inform remote routers of the failure. The remote routers initially tunnel affected traffic to the nearest PLR, using tunnels which are unaffected by the failure. Once the forwarding state for normal shortest path routing has converged, the remote routers return the traffic to shortest path forwarding. MRT-FRR is relevant for Nearside Tunneling for the following reason. The process of tunneling traffic to the PLRs and waiting a sufficient amount of time for IGP forwarding state convergence with Nearside Tunneling means that traffic will generally be relying on the local repair at the PLR for longer than it would in the absence of Nearside Tunneling. Since MRT-FRR provides 100% coverage for single link and node failure, it may be an attractive option to provide the local repair paths when Nearside Tunneling is deployed.

MRT-FRR is also relevant for the Farside Tunneling micro-loop prevention technique. In Farside Tunneling, remote routers tunnel traffic affected by a failure to a node downstream of the failure with respect to traffic destination. This node can be viewed as being on the farside of the failure with respect to the node initiating the tunnel. Note that the discussion of Farside Tunneling in [RFC5715] focuses on the case where the farside node is immediately adjacent to a failed link or node. However, the farside node may be any node downstream of the failure with respect to traffic destination, including the destination itself. The tunneling mechanism used to reach the farside node must be unaffected by the failure. The alternative forwarding paths created by MRT-FRR have the potential to be used to forward traffic from the remote routers upstream of the failure all the way to the destination. In the event of failure, either the MRT-Red or MRT-Blue path from the remote upstream router to the destination is guaranteed to avoid a link failure or inferred node failure. The MRT forwarding paths are also guaranteed to not be subject to micro-loops because they are locked to the topology before the failure.

We note that the computations in [I-D.ietf-rtgwg-mrt-frr-algorithm] address the case of a PLR adjacent to a failure determining which choice of MRT-Red or MRT-Blue will avoid a failed link or node. More computation may be required for an arbitrary remote upstream router to determine whether to choose MRT-Red or MRT-Blue for a given destination and failure.

12.2. MRT Recalculation for the Default MRT Profile

This section describes how the MRT recalculation SHOULD be performed for the Default MRT Profile. This is intended to support FRR applications. Other approaches are possible, but they are not specified in this document.

When a failure event happens, traffic is put by the PLRs onto the MRT topologies. After that, each router recomputes its shortest path tree (SPT) and moves traffic over to that. Only after all the PLRs have switched to using their SPTs and traffic has drained from the MRT topologies should each router install the recomputed MRTs into the FIBs.

At each router, therefore, the sequence is as follows:

1. Receive failure notification
2. Recompute SPT.
3. Install the new SPT in the FIB.
4. If the network was stable before the failure occurred, wait a configured (or advertised) period for all routers to be using their SPTs and traffic to drain from the MRTs.
5. Recompute MRTs.
6. Install new MRTs in the FIB.

While the recomputed MRTs are not installed in the FIB, protection coverage is lowered. Therefore, it is important to recalculate the MRTs and install them quickly.

New protocol extensions for advertising the time needed to recompute shortest path routes and install them in the FIB will be defined elsewhere.

13. Implementation Status

[RFC Editor: please remove this section prior to publication.]

This section records the status of known implementations of the protocol defined by this specification at the time of posting of this Internet-Draft, and is based on a proposal described in [RFC6982]. The description of implementations in this section is intended to assist the IETF in its decision processes in progressing drafts to RFCs. Please note that the listing of any individual implementation here does not imply endorsement by the IETF. Furthermore, no effort has been spent to verify the information presented here that was supplied by IETF contributors. This is not intended as, and must not be construed to be, a catalog of available implementations or their features. Readers are advised to note that other implementations may exist.

According to [RFC6982], "this will allow reviewers and working groups to assign due consideration to documents that have the benefit of running code, which may serve as evidence of valuable experimentation and feedback that have made the implemented protocols more mature. It is up to the individual working groups to use this information as they see fit".

Juniper Networks Implementation

- o Organization responsible for the implementation: Juniper Networks
- o Implementation name: MRT-FRR
- o Implementation description: MRT-FRR using OSPF as the IGP has been implemented and verified.
- o The implementation's level of maturity: prototype
- o Protocol coverage: This implementation of the MRT-FRR includes Island identification, GADAG root selection, MRT Lowpoint algorithm, augmentation of GADAG with additional links, and calculation of MRT transit next-hops alternate next-hops based on draft "draft-ietf-rtgwg-mrt-frr-algorithm-00". This implementation also includes the M-bit in OSPF based on "draft-atlas-ospf-mrt-01" as well as LDP MRT Capability based on "draft-atlas-mpls-ldp-mrt-00".
- o Licensing: proprietary

- o Implementation experience: Implementation was useful for verifying functionality and lack of gaps. It has also been useful for improving aspects of the algorithm.
- o Contact information: akatlas@juniper.net, shraddha@juniper.net, kishoret@juniper.net

Huawei Technology Implementation

- o Organization responsible for the implementation: Huawei Technology Co., Ltd.
- o Implementation name: MRT-FRR and IS-IS extensions for MRT.
- o Implementation description: The MRT-FRR using IS-IS extensions for MRT and LDP multi-topology have been implemented and verified.
- o The implementation's level of maturity: prototype
- o Protocol coverage: This implementation of the MRT algorithm includes Island identification, GADAG root selection, MRT Lowpoint algorithm, augmentation of GADAG with additional links, and calculation of MRT transit next-hops alternate next-hops based on draft "draft-enyedi-rtgwg-mrt-frr-algorithm-03". This implementation also includes IS-IS extension for MRT based on "draft-li-mrt-00".
- o Licensing: proprietary
- o Implementation experience: It is important produce a second implementation to verify the algorithm is implemented correctly without looping. It is important to verify the IS-IS extensions work for MRT-FRR.
- o Contact information: lizhenbin@huawei.com, eric.wu@huawei.com

14. Operational Considerations

The following aspects of MRT-FRR are useful to consider when deploying the technology in different operational environments and network topologies.

14.1. Verifying Forwarding on MRT Paths

The forwarding paths created by MRT-FRR are not used by normal (non-FRR) traffic. They are only used to carry FRR traffic for a short period of time after a failure has been detected. It is RECOMMENDED that an operator proactively monitor the MRT forwarding paths in

order to be certain that the paths will be able to carry FRR traffic when needed. Therefore, an implementation SHOULD provide an operator with the ability to test MRT paths with Operations, Administration, and Maintenance (OAM) traffic. For example, when MRT paths are realized using LDP labels distributed for topology-scoped FECs, an implementation can use the MPLS ping and traceroute as defined in [RFC4379] and extended in [RFC7307] for topology-scoped FECs.

14.2. Traffic Capacity on Backup Paths

During a fast-reroute event initiated by a PLR in response to a network failure, the flow of traffic in the network will generally not be identical to the flow of traffic after the IGP forwarding state has converged, taking the failure into account. Therefore, even if a network has been engineered to have enough capacity on the appropriate links to carry all traffic after the IGP has converged after the failure, the network may still not have enough capacity on the appropriate links to carry the flow of traffic during a fast-reroute event. This can result in more traffic loss during the fast-reroute event than might otherwise be expected.

Note that there are two somewhat distinct aspects to this phenomenon. The first is that the path from the PLR to the destination during the fast-reroute event may be different from the path after the IGP converges. In this case, any traffic for the destination that reaches the PLR during the fast-reroute event will follow a different path from the PLR to the destination than will be followed after IGP convergence.

The second aspect is that the amount of traffic arriving at the PLR for affected destinations during the fast-reroute event may be larger than the amount of traffic arriving at the PLR for affected destinations after IGP convergence. Immediately after a failure, any non-PLR routers that were sending traffic to the PLR before the failure will continue sending traffic to the PLR, and that traffic will be carried over backup paths from the PLR to the destinations. After IGP convergence, upstream non-PLR routers may direct some traffic away from the PLR.

In order to reduce or eliminate the potential for transient traffic loss due to inadequate capacity during fast-reroute events, an operator can model the amount of traffic taking different paths during a fast-reroute event. If it is determined that there is not enough capacity to support a given fast-reroute event, the operator can address the issue either by augmenting capacity on certain links or modifying the backup paths themselves.

The MRT Lowpoint algorithm produces a pair of diverse paths to each destination. These paths are generated by following the directed links on a common GADAG. The decision process for constructing the GADAG in the MRT Lowpoint algorithm takes into account individual IGP link metrics. At any given node, links are explored in order from lowest IGP metric to highest IGP metric. Additionally, the process for constructing the MRT-Red and Blue trees uses SPF traversals of the GADAG. Therefore, the IGP link metric values affect the computed backup paths. However, adjusting the IGP link metrics is not a generally applicable tool for modifying the MRT backup paths. Achieving a desired set of MRT backup paths by adjusting IGP metrics while at the same time maintaining the desired flow of traffic along the shortest paths is not possible in general.

MRT-FRR allows an operator to exclude a link from the MRT Island, and thus the GADAG, by advertising it as MRT-Ineligible. Such a link will not be used on the MRT forwarding path for any destination. Advertising links as MRT-Ineligible is the main tool provided by MRT-FRR for keeping backup traffic off of lower bandwidth links during fast-reroute events.

Note that all of the backup paths produced by the MRT Lowpoint algorithm are closely tied to the common GADAG computed as part of that algorithm. Therefore, it is generally not possible to modify a subset of paths without affecting other paths. This precludes more fine-grained modification of individual backup paths when using only paths computed by the MRT Lowpoint algorithm.

However, it may be desirable to allow an operator to use MRT-FRR alternates together with alternates provided by other FRR technologies. A policy-based alternate selection process can allow an operator to select the best alternate from those provided by MRT and other FRR technologies. As a concrete example, it may be desirable to implement a policy where a downstream LFA (if it exists for a given failure mode and destination) is preferred over a given MRT alternate. This combination gives the operator the ability to affect where traffic flows during a fast-reroute event, while still producing backup paths that use no additional labels for LDP traffic and will not loop under multiple failures. This and other choices of alternate selection policy can be evaluated in the context of their effect on fast-reroute traffic flow and available capacity, as well as other deployment considerations.

Note that future documents may define MRT profiles in addition to the default profile defined here. Different MRT profiles will generally produce alternate paths with different properties. An implementation may allow an operator to use different MRT profiles instead of or in addition to the default profile.

14.3. MRT IP Tunnel Loopback Address Management

As described in Section 6.1.2, if an implementation uses IP tunneling as the mechanism to realize MRT forwarding paths, each node must advertise an MRT-Red and an MRT-Blue loopback address. These IP addresses must be unique within the routing domain to the extent that they do not overlap with each other or with any other routing table entries. It is expected that operators will use existing tools and processes for managing infrastructure IP addresses to manage these additional MRT-related loopback addresses.

14.4. MRT-FRR in a Network with Degraded Connectivity

Ideally, routers in a service provider network using MRT-FRR will be initially deployed in a 2-connected topology, allowing MRT-FRR to find completely diverse paths to all destinations. However, a network can differ from an ideal 2-connected topology for many possible reasons, including network failures and planned maintenance events.

MRT-FRR is designed to continue to function properly when network connectivity is degraded. When a network contains cut-vertices or cut-links dividing the network into different 2-connected blocks, MRT-FRR will continue to provide completely diverse paths for destinations within the same block as the PLR. For a destination in a different block from the PLR, the redundant paths created by MRT-FRR will be link and node diverse within each block, and the paths will only share links and nodes that are cut-links or cut-vertices in the topology.

If a network becomes partitioned with one set of routers having no connectivity to another set of routers, MRT-FRR will function independently in each set of connected routers, providing redundant paths to destinations in same set of connected routers as a given PLR.

14.5. Partial Deployment of MRT-FRR in a Network

A network operator may choose to deploy MRT-FRR only on a subset of routers in an IGP area. MRT-FRR is designed to accommodate this partial deployment scenario. Only routers that advertise support for a given MRT profile will be included in a given MRT Island. For a PLR within the MRT Island, MRT-FRR will create redundant forwarding paths to all destinations with the MRT Island using maximally redundant trees all the way to those destinations. For destinations outside of the MRT Island, MRT-FRR creates paths to the destination which use forwarding state created by MRT-FRR within the MRT Island and shortest path forwarding state outside of the MRT Island. The

paths created by MRT-FRR to non-Island destinations are guaranteed to be diverse within the MRT Island (if topologically possible).

However, the part of the paths outside of the MRT Island may not be diverse.

15. Acknowledgements

The authors would like to thank Mike Shand for his valuable review and contributions.

The authors would like to thank Joel Halpern, Hannes Gredler, Ted Qian, Kishore Tiruveedhula, Shraddha Hegde, Santosh Esale, Nitin Bahadur, Harish Sitaraman, Raveendra Torvi, Anil Kumar SN, Bruno Decraene, Eric Wu, Janos Farkas, Rob Shakir, Stewart Bryant, and Alvaro Retana for their suggestions and review.

16. IANA Considerations

IANA is requested to create a registry entitled "MRT Profile Identifier Registry". The range is 0 to 255. The Default MRT Profile defined in this document has value 0. Values 1-200 are allocated by Standards Action. Values 201-220 are for Experimental Use. Values 221-254 are for Private Use. Value 255 is reserved for future registry extension. (The allocation and use policies are described in [RFC5226].)

The initial registry is shown below.

Value	Description	Reference
-----	-----	-----
0	Default MRT Profile	[This draft]
1-200	Unassigned	
201-220	Experimental Use	
221-254	Private Use	
255	Reserved (for future registry extension)	

The MRT Profile Identifier Registry is a new registry in the IANA Matrix. Following existing conventions, <http://www.iana.org/protocols> should display a new header entitled "Maximally Redundant Tree (MRT) Parameters". Under that header, there should be an entry for "MRT Profile Identifier Registry" with a link to the registry itself at <http://www.iana.org/assignments/mrt-parameters/mrt-parameters.xhtml#mrt-profile-registry>.

17. Security Considerations

In general, MRT forwarding paths do not follow shortest paths. The transit forwarding state corresponding to the MRT paths is created during normal operations (before a failure occurs). Therefore, a malicious packet with an appropriate header injected into the network from a compromised location would be forwarded to a destination along a non-shortest path. When this technology is deployed, a network security design should not rely on assumptions about potentially malicious traffic only following shortest paths.

It should be noted that the creation of non-shortest forwarding paths is not unique to MRT.

MRT-FRR requires that routers advertise information used in the formation of MRT backup paths. While this document does not specify the protocol extensions used to advertise this information, we discuss security considerations related to the information itself. Injecting false MRT-related information could be used to direct some MRT backup paths over compromised transmission links. Combined with the ability to generate network failures, this could be used to send traffic over compromised transmission links during a fast-reroute event. In order to prevent this potential exploit, a receiving router needs to be able to authenticate MRT-related information that claims to have been advertised by another router.

18. Contributors

Robert Kebler
Juniper Networks
10 Technology Park Drive
Westford, MA 01886
USA
Email: rkebler@juniper.net

Andras Csaszar
Ericsson
Konyves Kalman krt 11
Budapest 1097
Hungary
Email: Andras.Csaszar@ericsson.com

Jeff Tantsura
Ericsson
300 Holger Way
San Jose, CA 95134
USA
Email: jeff.tantsura@ericsson.com

Russ White
VCE
Email: russw@riw.us

19. References

19.1. Normative References

- [I-D.ietf-rtgwg-mrt-frr-algorithm]
Envedi, G., Csaszar, A., Atlas, A., Bowers, C., and A. Gopalan, "Algorithms for computing Maximally Redundant Trees for IP/LDP Fast- Reroute", draft-ietf-rtgwg-mrt-frr-algorithm-06 (work in progress), October 2015.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<http://www.rfc-editor.org/info/rfc2119>>.
- [RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, DOI 10.17487/RFC5226, May 2008, <<http://www.rfc-editor.org/info/rfc5226>>.

- [RFC7307] Zhao, Q., Raza, K., Zhou, C., Fang, L., Li, L., and D. King, "LDP Extensions for Multi-Topology", RFC 7307, DOI 10.17487/RFC7307, July 2014, <<http://www.rfc-editor.org/info/rfc7307>>.

19.2. Informative References

- [EnyediThesis] Enyedi, G., "Novel Algorithms for IP Fast Reroute", Department of Telecommunications and Media Informatics, Budapest University of Technology and Economics Ph.D. Thesis, February 2011, <http://timon.tmit.bme.hu/theses/thesis_book.pdf>.
- [I-D.atlas-rtgwg-mrt-mc-arch] Atlas, A., Kebler, R., Wijnands, I., Csaszar, A., and G. Envedi, "An Architecture for Multicast Protection Using Maximally Redundant Trees", draft-atlas-rtgwg-mrt-mc-arch-02 (work in progress), July 2013.
- [RFC2328] Moy, J., "OSPF Version 2", STD 54, RFC 2328, DOI 10.17487/RFC2328, April 1998, <<http://www.rfc-editor.org/info/rfc2328>>.
- [RFC4379] Kompella, K. and G. Swallow, "Detecting Multi-Protocol Label Switched (MPLS) Data Plane Failures", RFC 4379, DOI 10.17487/RFC4379, February 2006, <<http://www.rfc-editor.org/info/rfc4379>>.
- [RFC5286] Atlas, A., Ed. and A. Zinin, Ed., "Basic Specification for IP Fast Reroute: Loop-Free Alternates", RFC 5286, DOI 10.17487/RFC5286, September 2008, <<http://www.rfc-editor.org/info/rfc5286>>.
- [RFC5331] Aggarwal, R., Rekhter, Y., and E. Rosen, "MPLS Upstream Label Assignment and Context-Specific Label Space", RFC 5331, DOI 10.17487/RFC5331, August 2008, <<http://www.rfc-editor.org/info/rfc5331>>.
- [RFC5340] Coltun, R., Ferguson, D., Moy, J., and A. Lindem, "OSPF for IPv6", RFC 5340, DOI 10.17487/RFC5340, July 2008, <<http://www.rfc-editor.org/info/rfc5340>>.
- [RFC5443] Jork, M., Atlas, A., and L. Fang, "LDP IGP Synchronization", RFC 5443, DOI 10.17487/RFC5443, March 2009, <<http://www.rfc-editor.org/info/rfc5443>>.

- [RFC5714] Shand, M. and S. Bryant, "IP Fast Reroute Framework", RFC 5714, DOI 10.17487/RFC5714, January 2010, <<http://www.rfc-editor.org/info/rfc5714>>.
- [RFC5715] Shand, M. and S. Bryant, "A Framework for Loop-Free Convergence", RFC 5715, DOI 10.17487/RFC5715, January 2010, <<http://www.rfc-editor.org/info/rfc5715>>.
- [RFC6976] Shand, M., Bryant, S., Previdi, S., Filsfils, C., Francois, P., and O. Bonaventure, "Framework for Loop-Free Convergence Using the Ordered Forwarding Information Base (oFIB) Approach", RFC 6976, DOI 10.17487/RFC6976, July 2013, <<http://www.rfc-editor.org/info/rfc6976>>.
- [RFC6981] Bryant, S., Previdi, S., and M. Shand, "A Framework for IP and MPLS Fast Reroute Using Not-Via Addresses", RFC 6981, DOI 10.17487/RFC6981, August 2013, <<http://www.rfc-editor.org/info/rfc6981>>.
- [RFC6982] Sheffer, Y. and A. Farrel, "Improving Awareness of Running Code: The Implementation Status Section", RFC 6982, DOI 10.17487/RFC6982, July 2013, <<http://www.rfc-editor.org/info/rfc6982>>.
- [RFC6987] Retana, A., Nguyen, L., Zinin, A., White, R., and D. McPherson, "OSPF Stub Router Advertisement", RFC 6987, DOI 10.17487/RFC6987, September 2013, <<http://www.rfc-editor.org/info/rfc6987>>.
- [RFC7490] Bryant, S., Filsfils, C., Previdi, S., Shand, M., and N. So, "Remote Loop-Free Alternate (LFA) Fast Reroute (FRR)", RFC 7490, DOI 10.17487/RFC7490, April 2015, <<http://www.rfc-editor.org/info/rfc7490>>.

Appendix A. Inter-level Forwarding Behavior for IS-IS

In the description below, we use the terms "Level-1-only interface", "Level-2-only interface", and "Level-1-and-Level-2 interface" to mean in interface which has formed only a Level-1 adjacency, only a Level-2 adjacency, or both Level-1 and Level-2 adjacencies. Note that IS-IS also defines the concept of areas. A router is configured with an IS-IS area identifier, and a given router may be configured with multiple IS-IS area identifiers. For an IS-IS Level-1 adjacency to form between two routers, at least one IS-IS area identifier must match. IS-IS Level-2 adjacencies do not require any area identifiers to match. The behavior described below does not explicitly refer to IS-IS area identifiers. However, IS-IS area identifiers will

indirectly affect the behavior by affecting the formation of Level-1 adjacencies.

First consider a packet destined to Z on MRT-Red or MRT-Blue received on a Level-1-only interface. If the best shortest path route to Z was learned from a Level-1 advertisement, then the packet should continue to be forwarded along MRT-Red or MRT-Blue. If instead the best route was learned from a Level-2 advertisement, then the packet should be removed from MRT-Red or MRT-Blue and forwarded on the shortest-path default forwarding topology.

Now consider a packet destined to Z on MRT-Red or MRT-Blue received on a Level-2-only interface. If the best route to Z was learned from a Level-2 advertisement, then the packet should continue to be forwarded along MRT-Red or MRT-Blue. If instead the best route was learned from a Level-1 advertisement, then the packet should be removed from MRT-Red or MRT-Blue and forwarded on the shortest-path default forwarding topology.

Finally, consider a packet destined to Z on MRT-Red or MRT-Blue received on a Level-1-and-Level-2 interface. This packet should continue to be forwarded along MRT-Red or MRT-Blue, regardless of which level the route was learned from.

An implementation may simplify the decision-making process above by using the interface of the next-hop for the route to Z to determine the level that the best route to Z was learned from. If the next-hop points out a Level-1-only interface, then the route was learned from a Level-1 advertisement. If the next-hop points out a Level-2-only interface, then the route was learned from a Level-2 advertisement. A next-hop that points out a Level-1-and-Level-2 interface does not provide enough information to determine the source of the best route. With this simplification, an implementation would need to continue forwarding along MRT-Red or MRT-Blue when the next-hop points out a Level-1-and-Level-2 interface. Therefore, a packet on MRT-Red or MRT-Blue going from Level-1 to Level-2 (or vice versa) that traverses a Level-1-and-Level-2 interface in the process will remain on MRT-Red or MRT-Blue. This simplification may not always produce the optimal forwarding behavior, but it does not introduce interoperability problems. The packet will stay on an MRT backup path longer than necessary, but it will still reach its destination.

Appendix B. General Issues with Area Abstraction

When a multi-homed prefix is connected in two different areas, it may be impractical to protect them without adding the complexity of explicit tunneling. This is also a problem for LFA and Remote-LFA.

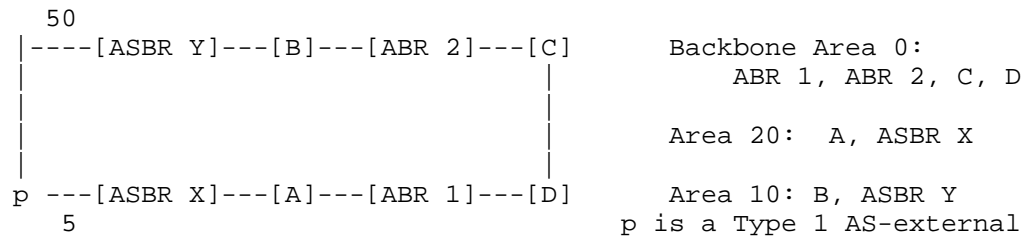


Figure 4: AS external prefixes in different areas

Consider the network in Figure 4 and assume there is a richer connective topology that isn't shown, where the same prefix is announced by ASBR X and ASBR Y which are in different non-backbone areas. If the link from A to ASBR X fails, then an MRT alternate could forward the packet to ABR 1 and ABR 1 could forward it to D, but then D would find the shortest route is back via ABR 1 to Area 20. This problem occurs because the routers, including the ABR, in one area are not yet aware of the failure in a different area.

The only way to get it from A to ASBR Y is to explicitly tunnel it to ASBR Y. If the traffic is unlabeled or the appropriate MPLS labels are known, then explicit tunneling MAY be used as long as the shortest-path of the tunnel avoids the failure point. In that case, A must determine that it should use an explicit tunnel instead of an MRT alternate.

Authors' Addresses

Alia Atlas
 Juniper Networks
 10 Technology Park Drive
 Westford, MA 01886
 USA

Email: akatlas@juniper.net

Chris Bowers
 Juniper Networks
 1194 N. Mathilda Ave.
 Sunnyvale, CA 94089
 USA

Email: cbowers@juniper.net

Gabor Sandor Enyedi
Ericsson
Konyves Kalman krt 11.
Budapest 1097
Hungary

Email: Gabor.Sandor.Enyedi@ericsson.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 29, 2013

Z. Li
N. Wu
Q. Zhao
Huawei Technologies
February 25, 2013

Routing Extension for Fast-Reroute Using Maximally Redundant Trees
draft-li-rtgwg-igp-ext-mrt-frr-01

Abstract

The document proposes the routing protocol extension and procedures to support fast reroute using maximally redundant trees.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on August 29, 2013.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect

to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
2. Terminology	3
3. MRT-FRR TLV Format	3
3.1. IS-IS MRT-FRR Sub-TLV Format	3
3.2. OSPF MRT FRR TLV Format	5
4. Elements of Procedure	7
4.1. IS-IS	7
4.1.1. Sending	7
4.1.2. Receiving	8
4.2. OSPF	9
4.2.1. Sending	9
4.2.2. Receiving	10
5. Backward Compatibility	10
6. IANA Considerations	11
6.1. IS-IS	11
6.2. OSPF	11
7. Security Considerations	11
8. Normative References	11
Authors' Addresses	12

1. Introduction

[I-D.ietf-rtgwg-mrt-frr-architecture] describes the architecture based on Maximally Redundant Trees (MRT) to provide 100% coverage for FRR of unicast traffic. Protocol extensions and considerations are also been proposed. The document defines the extension of routing protocols for fast reroute using maximally redundant trees. The detailed procedures are defined based the extension.

2. Terminology

GADAG: Generalized ADAG - a graph that is the combination of the ADAGs of all blocks.

Maximally Redundant Trees (MRT): A pair of trees where the path from any node X to the root R along the first tree and the path from the same node X to the root along the second tree share the minimum number of nodes and the minimum number of links. Each such shared node is a cut-vertex. Any shared links are cut-links. Any RT is an MRT but many MRTs are not RTs.

3. MRT-FRR TLV Format

3.1. IS-IS MRT-FRR Sub-TLV Format

IS-IS MRT FRR sub-TLV is defined to advertise necessary information related with MRT FRR. It is an optional sub-TLV which can be advertised in the router capability TLV([RFC4971]) . The information has only level-wide scope. If there is no MRT FRR sub-TLV advertised, that router should be seen as that it does not support MRT FRR.

The IS-IS MRT FRR sub-TLV is composed of 1 octet for the type, 1 octet specifying the TLV length and a value field. The IS-IS MRT FRR sub-TLV format (Figure 1) is as follows:

TYPE: TBD

LENGTH: 12

	No. of Octets
+-----+ R R R R Primary MT ID	2
+-----+ R R R R Blue MRT MT ID	2
+-----+ R R R R Red MRT MT ID	2
+-----+ MRT Capabilities Available	2
+-----+ MRT Algorithm ID	1
+-----+ MRT Fd Mechanism	1
+-----+ GADAG Root Election Priority	2
+-----+	

Figure 1 IS-IS MRT FRR Sub-TLV Format

The IS-IS MRT FRR sub-TLV is made of the following fields:

-- Primary MT-ID: This specifies the MT-ID of the primary topology, 12 bits.

-- Blue MRT MT-ID: This specifies the MT-ID to be associated with the Blue MRT forwarding topology. It is needed for use in signaling. All routers in the MRT Island MUST agree on a value, 12 bits.

-- Red MRT MT-ID: This specifies the MT-ID to be associated with the Red MRT forwarding topology. It is needed for use in signaling. All routers in the MRT Island MUST agree on a value, 12 bits.

-- MRT Capabilities Available: This specifies the set of capabilities that the router can support.

```

+-----+
|0|1|2|3|4|5|6|*| Reserved |
+-----+

```

```

Bit0 - MRT-BIT
Bit1 - IP-BIT
Bit2 - LDP-BIT
Bit3 - PIM-BIT
Bit4 - PIMG-BIT
Bit5 - mLDP-BIT
Bit6 - mLDPG-BIT

```

-- MRT Algorithm ID: This identifies the particular MRT algorithm used by the router. By having an Algorithm ID, it is possible to

change the algorithm used or use different ones in different networks. It may be desirable to advertise a list ordered by preference to allow transitions.

```

+---+---+---+---+---+
|0|1|2|*|*|*|*|*|
+---+---+---+---+---+

```

Bit0 - LP-BIT
 Bit1 - SPF-BIT
 Bit2 - HYBRID-BIT

-- MRT Fd Mechanism: This specifies which forwarding mechanisms the router supports. If IP-in-IPv4 or IP-in-IPv6 is used as forwarding mechanisms for IP, Red MRT Loopback Address and Blue MRT Loopback Address should be advertised by the Multi-Topology Reachable IPv4/IPv6 Prefixes TLV ([RFC5120]).

```

+---+---+---+---+---+
|0|*|2|3|4|*|*|*|
+---+---+---+---+---+

```

Bit0: LDP Destination-Topology Label
 Bit2: IP-in-IPv4
 Bit3: IP-in-IPv6
 Bit4: Encode MT-ID in Labels

-- GADAG Root Election Priority: This specifies the priority of the router for being used as the GADAG root of its island. A GADAG root is elected from the set of routers with the highest priority; ties are broken based upon highest System ID. The sensitivity of the MRT Algorithms to GADAG root selection is still being evaluated. This provides the network operator with a knob to force particular GADAG root selection.

3.2. OSPF MRT FRR TLV Format

OSPF MRT FRR TLV is defined to advertise necessary information related with MRT FRR. It is an optional TLV which can be advertised in the OSPF router information LSA([RFC4970]) . The information has only area-wide scope. If there is no MRT FRR TLV advertised, that router should be seen as that it does not support MRT FRR.

The OSPF MRT FRR TLV has the following format:

TYPE: TBD

LENGTH: 12

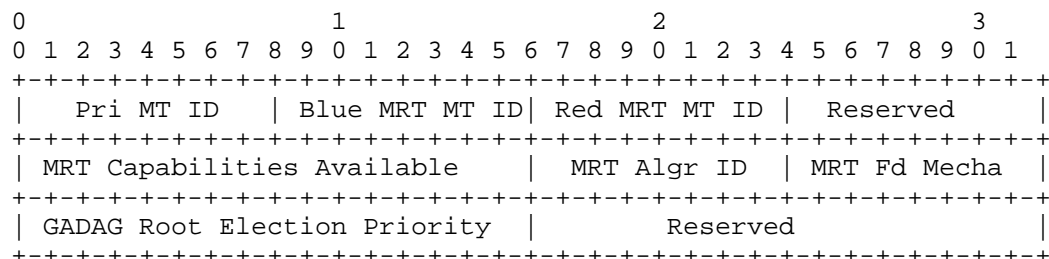


Figure 2 OSPF MRT FRR TLV Format

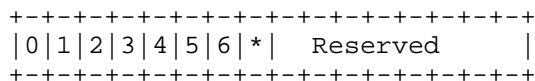
The OSPF MRT FRR TLV is made of the following fields:

-- Pri MT-ID: This specifies the MT-ID of the primary topology. It is a 7-bit-field since AS-External-LSAs use the high-order bit in the MT-ID field (E-bit) for the external metric-type.

-- Blue MRT MT-ID: This specifies the MT-ID to be associated with the Blue MRT forwarding topology. It is needed for use in signaling. All routers in the MRT Island MUST agree on a value, 7 bits.

-- Red MRT MT-ID: This specifies the MT-ID to be associated with the Red MRT forwarding topology. It is needed for use in signaling. All routers in the MRT Island MUST agree on a value, 7 bits.

-- MRT Capabilities Available: This specifies the set of capabilities that the router can support.



Bit0 - MRT-BIT
 Bit1 - IP-BIT
 Bit2 - LDP-BIT
 Bit3 - PIM-BIT
 Bit4 - PIMG-BIT
 Bit5 - mLDP-BIT
 Bit6 - mLDPG-BIT

-- MRT Algr ID: This identifies the particular MRT algorithm used by the router. By having an Algorithm ID, it is possible to change the algorithm used or use different ones in different networks. It may be desirable to advertise a list ordered by preference to allow transitions.

```

+--+--+--+--+--+--+--+
|0|1|2|*|*|*|*|*|
+--+--+--+--+--+--+--+

```

Bit0 - LP-BIT
 Bit1 - SPF-BIT
 Bit2 - HYBRID-BIT

-- MRT Fd Mecha: This specifies which forwarding mechanisms the router supports. If IP-in-IPv4 or IP-in-IPv6 are used as forwarding mechanisms for IP, Red MRT Loopback Address and Blue MRT Loopback Address should be advertised by the Multi-Topology Reachable IPv4/IPv6 Prefixes TLV ([RFC4915]).

```

+--+--+--+--+--+--+--+
|0|*|2|3|4|*|*|*|
+--+--+--+--+--+--+--+

```

Bit0: LDP Destination-Topology Label
 Bit2: IP-in-IPv4
 Bit3: IP-in-IPv6
 Bit4: Encode MT-ID in Labels

-- GADAG Root Election Priority: This specifies the priority of the router for being used as the GADAG root of its island. A GADAG root is elected from the set of routers with the highest priority; ties are broken based upon highest Router ID. The sensitivity of the MRT Algorithms to GADAG root selection is still being evaluated. This provides the network operator with a knob to force particular GADAG root selection.

4. Elements of Procedure

4.1. IS-IS

4.1.1. Sending

MRT FRR sub-TLV is encapsulated in the Router Capability TLV and advertised through LSP PDU in the level-wide. MRT FRR sub-TLV is an optional TLV. If the router cannot support MRT FRR, it MUST NOT send the sub-TLV. Since the advertisement scope of the MRT sub-TLV is level-wide, when it is advertised the D-Bit and S-Bit of the Router Capability TLV MUST be set as 0. If other sub-TLVs in the Router Capability TLV need different values for the two bits, there MUST be an independent Router Capability TLV for the MRT FRR sub-TLV.

If the router can support multiple MRT FRR instances, there can be multiple MRT FRR sub-TLVs to be advertised. In these sub-TLVs and

there are different primary MT-IDs and associated Red/Blue MT-IDs. MT-IDs advertised by the MRT FRR sub-TLV MUST NOT be the reserved values for MT-ID([RFC5120]). In one MRT FRR sub-TLV, the Blue MT-ID MUST be different from the Red MT-ID.

MRT FRR sub-TLV MUST advertise the actual MRT capabilities, MRT algorithms and MRT forwarding mechanisms that the router can support. The corresponding bit MUST be set as 1 if it is supported. Otherwise, it MUST be set as 0.

GADAG Root Election Priority is a 16-bit unsigned integer which default value is set to 0x8000. The higher numerical value means the higher priority. GADAG Root Election is ordered with the priority and the IS-IS system ID is used as the tiebreaker. That is, if the priority is the same, the router with higher IS-IS system ID will be chosen. When a new MRT-capable router is added and its priority is higher than the current root, the MRT island will recalculate GADAG and new Blue/Red next hops for each prefix in the primary topology. If the current root fails, the new root will be re-elected and MRT calculation will be done according to the new root. Routers that are marked as overloaded([RFC3787]) is not qualified as candidate for root selection.

When MRT related information is changed in the router or existing IS-IS LSP mechanisms are triggered for refresh or update, MRT sub-TLV MUST be advertised with LSP.

4.1.2. Receiving

MRT capability SHOULD NOT affect the peer setup and the routing calculation of the default SPF topology.

MRT FRR sub-TLV should be validated like other sub-TLVs when received. MRT FRR sub-TLV is also taken for the checksum calculation and authentication.

If MRT FRR sub-TLV is received and the D-Bit and S-Bit of the router capability TLV may be not 0, MRT FRR sub-TLV MUST NOT leak to other levels. If the receiver router can not support MRT, it MUST ignore MRT FRR sub-TLV.

If multiple MRT FRR sub-TLVs are received in one LSP, it means multiple MRT instances are supported by the sender router. If the MT-ID conflict is found in the multiple MRT FRR sub-TLVs, the associated sub-TLVs MUST be ignored.

The MRT capability field, the MRT algorithm field and the MRT forwarding mechanism field MUST NOT be 0. If one of these field is

0, the sub-TLV MUST be ignored.

According to the group {primary MT-ID, the Red MRT MT-ID and the Blue MRT MT-ID} identified by the received MRT FRR sub-TLV, the receiver router will take the MRT capability for intersection. If there is no intersection, the router will stop processing for the group. For the MRT algorithm, the receiver router will also take it for intersection. If there is no intersection, the router will stop processing. For the MRT forwarding mechanism, the receiver router not only check if there is intersection, but also check if the intersection found can satisfy the requirement of the MRT capability intersection.

After the intersection is found for the group {primary MT-ID, the Red MRT MT-ID and the Blue MRT MT-ID}, the receiver router will elect the root node according to the GADAG Root Election Priority. If there are updates about the priority for existing MRT FRR sub-TLV or the new MRT FRR sub-TLV is received or the current root node fails, the receiver will recalculate GADAG and new Blue/Red next hops for each prefix in the primary topology.

4.2. OSPF

4.2.1. Sending

MRT FRR TLV is encapsulated in the router information LSA whose opaque type is 10 advertised in the area-wide. MRT FRR TLV is an optional TLV. If the router can not support MRT FRR, it MUST NOT send the TLV.

If the router can support multiple MRT FRR instances, there can be multiple MRT FRR TLVs to be advertised. In these TLVs and there are different primary MT-IDs and associated Red/Blue MT-IDs. MT-IDs advertised by the MRT FRR TLV MUST NOT be the reserved values for MT-ID([RFC4915]). In one MRT FRR TLV, the Blue MT-ID MUST be different from the Red MT-ID.

MRT FRR TLV MUST advertise the actual MRT capabilities, MRT algorithms and MRT forwarding mechanisms that the router can support. The corresponding bit MUST be set as 1 if it is supported. Otherwise, it MUST be set 0.

GADAG Root Election Priority is a 16-bit unsigned integer which default value is set to 0x8000. The higher numerical value means the higher priority. GADAG Root Election is ordered with the priority and the OSPF Router ID is used as the tiebreaker. That is, if the priority is the same, the router with higher OSPF Router ID will be chosen. When a new MRT-capable router is added and its priority is

higher than the current root, the MRT island will recalculate GADAG and new Blue/Red next hops for each prefix in the primary topology. If the current root fails, the new root will be re-elected and MRT calculation will be done according to the new root. Routers that are marked as stub router([RFC3137]) are not qualified as candidate for root selection.

When MRT related information is changed in the router or existing OSPF LSA mechanisms are triggered for refresh or update, MRT TLV MUST be advertised with LSA.

4.2.2. Receiving

MRT capability SHOULD NOT affect the neighbor setup and the routing calculation of the default SPF topology.

MRT FRR TLV should be validated like other TLVs when received. MRT FRR TLV is also taken for the checksum calculation and authentication.

The MRT capability field, the MRT algorithm field and the MRT forwarding mechanism field MUST NOT be 0. If one of these field is 0, the TLV MUST be ignored.

According to the group {primary MT-ID, the Red MRT MT-ID and the Blue MRT MT-ID} identified by the received MRT FRR TLV, the receiver router will take the MRT capability for intersection. If there is no intersection, the router will stop processing for the group. For the MRT algorithm, the receiver router will also take it for intersection. If there is no intersection, the router will stop processing. For the MRT forwarding mechanism, the receiver router not only check if there is intersection, but also check if the intersection found can satisfy the requirement of the MRT capability intersection.

After the intersection is found for the group {primary MT-ID, the Red MRT MT-ID and the Blue MRT MT-ID}, the receiver router will elect the root node according to the GADAG Root Election Priority. If there are updates about the priority for existing MRT FRR TLV or the new MRT FRR TLV is received or the current root node fails, the receiver will recalculate GADAG and new Blue/Red next hops for each prefix in the primary topology.

5. Backward Compatibility

The MRT FRR TLVs defined in this document do not introduce any interoperability issue. For OSPF, a router not supporting the MRT

FRR TLV SHOULD just silently ignore the TLV as specified in [RFC2370]. For an IS-IS, a router not supporting the MRT FRR sub-TLV SHOULD just silently ignore the sub-TLV.

6. IANA Considerations

6.1. IS-IS

This document introduces a new sub-TLV for IS-IS. The type is to be determined by IANA.

6.2. OSPF

This document introduces a new TLV for OSPF. The type is to be determined by IANA.

7. Security Considerations

This routing extension is not currently believed to introduce new security concerns.

8. Normative References

[I-D.enyedi-rtgwg-mrt-frr-algorithm]

Atlas, A., Envedi, G., Csaszar, A., and A. Gopalan,
"Algorithms for computing Maximally Redundant Trees for
IP/LDP Fast- Reroute",
draft-enyedi-rtgwg-mrt-frr-algorithm-02 (work in
progress), October 2012.

[I-D.ietf-rtgwg-mrt-frr-architecture]

Atlas, A., Kebler, R., Envedi, G., Csaszar, A.,
Konstantynowicz, M., White, R., and M. Shand, "An
Architecture for IP/LDP Fast-Reroute Using Maximally
Redundant Trees", draft-ietf-rtgwg-mrt-frr-architecture-01
(work in progress), March 2012.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate
Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3137] Retana, A., Nguyen, L., White, R., Zinin, A., and D.
McPherson, "OSPF Stub Router Advertisement", RFC 3137,
June 2001.

[RFC3787] Parker, J., "Recommendations for Interoperable IP Networks

using Intermediate System to Intermediate System (IS-IS)", RFC 3787, May 2004.

- [RFC4915] Psenak, P., Mirtorabi, S., Roy, A., Nguyen, L., and P. Pillay-Esnault, "Multi-Topology (MT) Routing in OSPF", RFC 4915, June 2007.
- [RFC4970] Lindem, A., Shen, N., Vasseur, JP., Aggarwal, R., and S. Shaffer, "Extensions to OSPF for Advertising Optional Router Capabilities", RFC 4970, July 2007.
- [RFC4971] Vasseur, JP., Shen, N., and R. Aggarwal, "Intermediate System to Intermediate System (IS-IS) Extensions for Advertising Router Information", RFC 4971, July 2007.
- [RFC5120] Przygienda, T., Shen, N., and N. Sheth, "M-ISIS: Multi Topology (MT) Routing in Intermediate System to Intermediate Systems (IS-ISs)", RFC 5120, February 2008.

Authors' Addresses

Zhenbin Li
Huawei Technologies
Huawei Bld., No.156 Beiqing Rd.
Beijing 100095
China

Email: lizhenbin@huawei.com

Nan Wu
Huawei Technologies
Huawei Bld., No.156 Beiqing Rd.
Beijing 100095
China

Email: eric.wu@huawei.com

Quintin Zhao
Huawei Technologies
125 Nagog Technology Park
Acton, MA 01719
US

Email: quintin.zhao@huawei.com

Network Working Group
Internet-Draft
Intended status: Informational
Expires: October 28, 2013

Zhenbin Li
Tao Zhou
Quintin Zhao
Huawei Technologies
Tianle Yang
China Mobile
April 26, 2013

Applicability of LDP Multi-Topology for Unicast Fast-reroute Using
Maximally Redundant Trees
draft-li-rtgwg-ldp-mt-mrt-frr-02

Abstract

In this document, procedures' considerations on the applicability of LDP MT for unicast fast-reroute using MRT are provided. The behaviors of label allocation and label forwarding entry setup with LDP Multi-Topology and MRT fast-reroute are described in details. Different application scenarios of the combining of the LDP multi-topology(MT) and unicast fast-reroute using Maximally Redundant Trees(MRT) are also analyzed.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on October 28, 2013.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
2. Terminology	3
3. Operation Procedures	4
3.1. Routing Calculation	4
3.2. Label Distribution	5
3.3. Forwarding Entry Creation	5
3.4. Switchover and Re-Convergence	5
3.5. Switchback	6
4. Application Scenario Analysis	6
4.1. 2-Connected Network	6
4.2. Non-2-Connected Network	10
4.3. Proxy Node	11
4.4. Inter-Area and Inter-AS	11
4.5. Partial Deployment	14
4.6. LDP over TE	15
4.7. IP-Only Network	16
5. Deployment Considerations	16
5.1. IGP MT and LDP MT	16
5.2. Simplified Provision	17
5.3. IGP Multi-process	18
5.4. Multiple IGP	21
5.5. Label Space	22
5.6. Proxy Egress	22
5.7. Policy Control	22
5.8. Resource Allocations	23
5.9. LDP DOD	23
6. IANA Considerations	23
7. Security Considerations	23
8. Acknowledgements	23
9. Normative References	23
Authors' Addresses	24

1. Introduction

[I-D.ietf-rtgwg-mrt-frr-architecture] describes the architecture based on Maximally Redundant Trees (MRT) to provide 100% coverage for fast-reroute of unicast traffic. LDP multi-topology [I-D.ietf-mpls-ldp-multi-topology] has been proposed to provide multi-topology-based unicast forwarding in the architecture. Guidance is provided for different application scenarios to improve the applicability. The analysis of the applicability of LDP MT for unicast fast-reroute using MRT is provided. The procedures are described and typical examples are provided based on LDP MT and MRT unicast FRR architecture.

When LDP MT is combined with MRT FRR, follow advantages can be achieved:

- o Provide 100% coverage for unicast traffic.
- o The complexity of the algorithm is moderate in $O(e)$ or $o(e + n \log n)$.
- o Co-deployment with LFA to provide better protection coverage.
- o Simplify operation and management with few additional configurations and states introduced.
- o Inherit procedures of LDP to achieve high scalability.
- o Propose no additional change on label forwarding behavior in the forwarding plane to facilitate incremental deployment.

2. Terminology

Some of terminologies defined in the [I-D.ietf-rtgwg-mrt-frr-architecture] are repeated here for the clarity of this document.

- o 2-connected: A graph that has no cut-vertices. This is a graph that requires two nodes to be removed before the network is partitioned.
- o 2-connected cluster: A maximal set of nodes that are 2-connected.
- o 2-edge-connected: A network graph where at least two links must be removed to partition the network.
- o cut-link: A link whose removal partitions the network. A cut-link by definition must be connected between two cut-vertices. If

there are multiple parallel links, then they are referred to as cut-links in this document if removing the set of parallel links would partition the network.

- o cut-vertex: A vertex whose removal partitions the network.
- o ECMP Equal cost multi-path: Where, for a particular destination D, multiple primary next-hops are used to forward traffic because there exist multiple shortest paths from S via different output layer-3 interfaces.
- o FIB Forwarding Information Base. The database used by the packet forwarder to determine what actions to perform on a packet.
- o LFA Loop-Free Alternate. A neighbor N, that is not a primary neighbor E, whose shortest path to the destination D does not go back through the router S. The neighbor N must meet the following condition: $\text{Distance_opt}(N, D) < \text{Distance_opt}(N, S) + \text{Distance_opt}(S, D)$
- o Maximally Redundant Trees (MRT): A pair of trees where the path from any node X to the root R along the first tree and the path from the same node X to the root along the second tree share the minimum number of nodes and the minimum number of links. Each such shared node is a cut-vertex. Any shared links are cut-links. Any RT is an MRT but many MRTs are not RTs.
- o Redundant Trees (RT): A pair of trees where the path from any node X to the root R along the first tree is node-disjoint with the path from the same node X to the root along the second tree. These can be computed in 2-connected graphs.
- o SPF Shortest Path First, e.g., Dijkstra's algorithm.
- o SPT Shortest path tree

3. Operation Procedures

3.1. Routing Calculation

IGP will flood information related with MRT FRR of each router and calculate routes for all destinations based on MRT. The details for the algorithm can refer to [I-D.enyedi-rtgwg-mrt-frr-algorithm]. For each destination, there are at least three routes that are associated with default topology, red topology and blue topology. The route of red topology or blue topology will be chosen as the secondary route. During the routing calculation, consistency of all nodes in the network must be kept. In order to achieve the consistence, following rules should be specified for the MRT calculation:

-- rules for choosing the root node;

-- rules for choosing the next-hop in the blue topology and the red topology.

Rules for choosing the secondary route can be determined locally if multiple routes exist. According to [I-D.ietf-rtgwg-mrt-frr-architecture], the secondary route derived from LFA calculation is preferred since it exists in the default topology. If there is no secondary route of LFA, the secondary route will be chosen in the blue topology or the red topology.

3.2. Label Distribution

When LDP MT is used for MRT fast-reroute, LDP will negotiate the MT Capability with LDP peer. Once IGP calculates routes for destinations based on MRT, LDP will advertise label mapping message with corresponding MT-ID for the specific FEC. There are at least three label bindings for each FEC that are associated with default topology, red topology and blue topology. We use `L_primary` for the label binding of the default topology, `L_blue` for the label binding of the blue topology and `L_red` for the label binding of the red topology.

MT-IDs, used in IGP and LDP, for the blue topology and the red topology can be specified administratively. In order to avoid inconsistency and simplify operation and management, we suggest MT-IDs should be reserved for MRT fast-reroute usage and the MT-IDs used in IP and LDP should be consistent.

3.3. Forwarding Entry Creation

LDP creates label forwarding entry for each FEC in different topologies. The route calculated based on MRT determines which label binding should be chosen for each FEC in a specific topology. For the default topology, the secondary label forwarding entry is determined by the secondary route in the blue topology or the red topology. Though the forwarding entry need be chosen according to MT information, there is not any MT information which should be processed in the forwarding plane so that the existing label forwarding mechanism can be reused well for MRT fast-reroute.

3.4. Switchover and Re-Convergence

When failure happens, the traffic can be switched to the secondary forwarding entry by forwarding plane within 50ms. The control plane will do the re-convergence process according to the link state change. During the course of re-convergence, the micro-loop may be

produced. The methods proposed by [RFC5714] can be used to prevent such micro-loop.

MRT fast-reroute calculation should be delayed. Thus the MRT topologies are carrying traffic and should not be disrupted until the new SPF routes are installed everywhere so that traffic is moved off of the MRT topologies. This way can prevent traffic loss to some extent. On the other hand, if failure happens again in the delayed period of MRT FRR calculation, it may cause traffic loss since the secondary route entry can not be guaranteed.

3.5. Switchback

When link failure or node failure recovers, IGP-LDP synchronization should be used for the default topology to prevent traffic loss. During the period of IGP-LDP synchronization, MRT FRR calculation can also be done. It will provide a best-effort protection if failure happens in the period.

4. Application Scenario Analysis

4.1. 2-Connected Network

In order to explain how LDP MT works for MRT FRR, we choose the following typical topology as an example. In the figure, (a) is the original topology, (b) is the blue topology calculated by MRT and (c) is the red topology calculated by MRT.

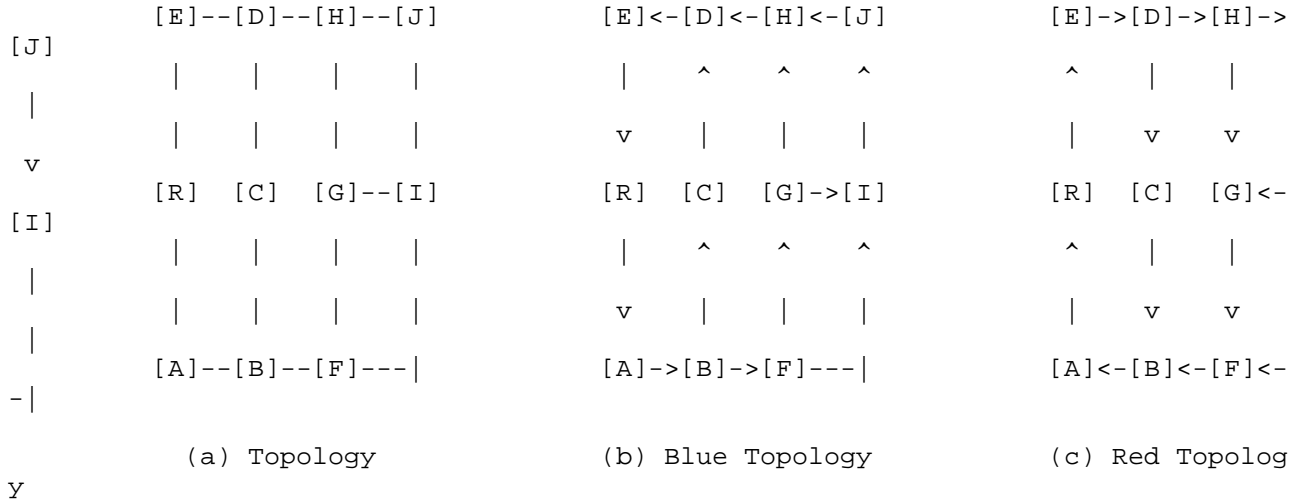


Figure 1: 2-Connected Network

According to the MRT calculation, for a specific destination H, there are following paths in different topologies for other nodes,

	Default Topology	Blue Topology	Red Topology
R	R->A->B->F->G->H	R->A->B->F->G->H	R->E->D->H
A	A->B->F->G->H	A->B->F->G->H	A->R->E->D->H
B	B->F->G->H	B->F->G->H	B->A->R->E->D->H

C	C->B->F->G->H	C->B->F->G->H	C->D->H
D	D->C->B->F->G->H	D->E->R->A->B->F	D->H
E	E->D->C->B->F->G->H	E->R->A->B->F->G->H	E->D->H
F	F->G->H	F->G->H	F->B->A->R->E->D->H
G	G->H	G->H	G->F->B->A->R->E->D->H
I	I->G->H	I->J->H	I->G->F->B->A->R->E->D->H
H			
J	J->H	J->H	J->I->G->F->B->A->R->E->D->H

Figure 2: Paths in Different Topologies for H

Note:

1. Assume that the metric of {E,R}, {D,H}, {R,C}, {G,I} and {F,I} is extreme high so that the route of the default topology is reasonable.

2. Assume tie-breaking rules determine that in blue topology the route from G to H chooses the path G->H instead of G->I->J->H.

3. Assume tie-breaking rules determine that in red topology the route from I to H chooses the path I->G->F->B->A->R->E->D->H instead of I->F->B->A->R->E->D->H.

4. For D node, both blue topology and red topology are available for the backup. The blue topology is preferred.

From the above calculation example, we can see that how the tie-breaking rule has to be applied when choose the nexthop in a specific topology and the topology which is used for the secondary route. For the reason of simplicity, there is no LFA calculation for the secondary route. If exists, it should be preferred.

We assume that labels are allocated in different topologies as the following figure.

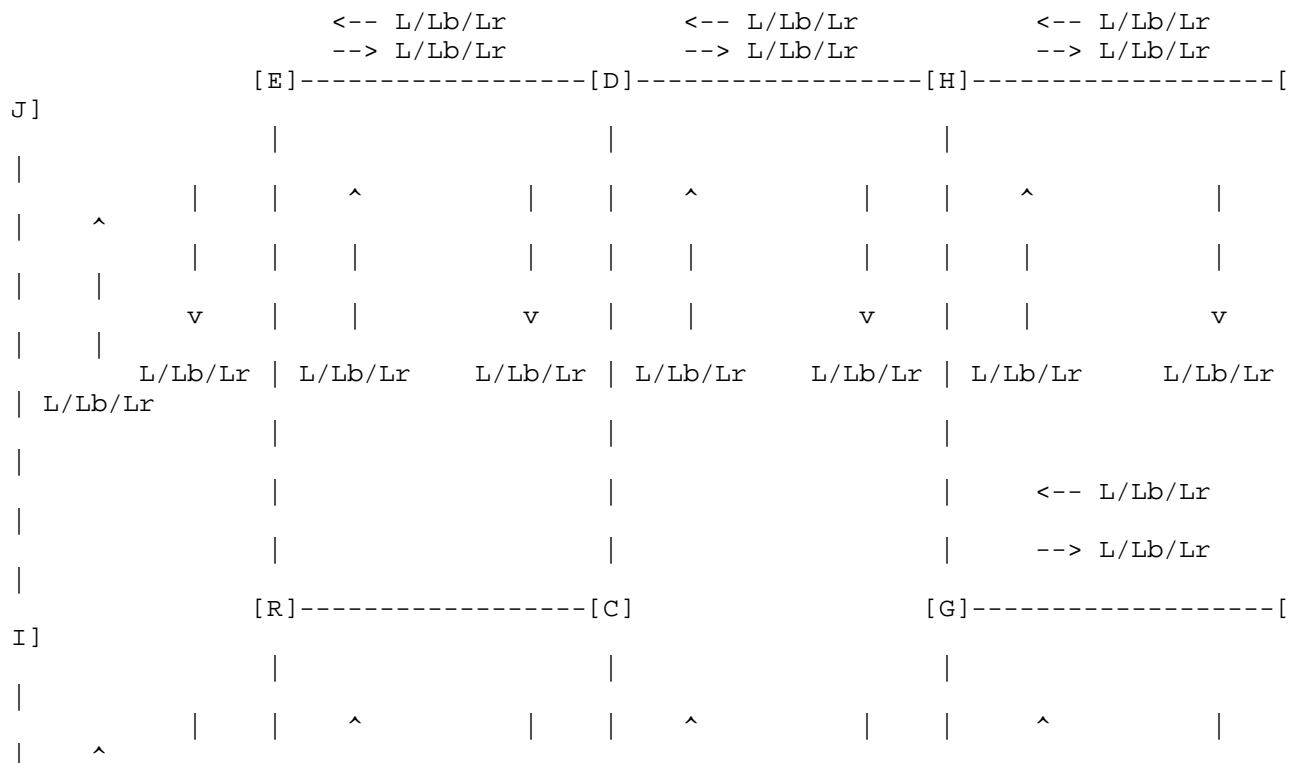




Figure 3: Label Allocation for LDP Multi-Topology

Note:

1. "<--" means the direction in which the label is distributed. For example, "<--" from D to E means that the label is distributed by D to E.
2. L means the label for H distributed in the default topology. Lb means the label for H distributed in the blue topology. Lr means the label for H distributed in the red topology.
3. L distributed by different nodes in the default topology does not mean they must be the same. This is also applied to Lb and Lr.

According to above MRT calculation result and label allocation for multi-topology, following forwarding entries will be installed for each node:

		Default Topology	Blue Topology	Red Topology
R	Ingress	--/L A /Lr E		
	Transit	L/L A /Lr E	Lb/Lb A /Lr E	Lr/Lr E
A	Ingress	--/L B /Lr R		
	Transit	L/L B /Lr R	Lb/Lb B /Lr R	Lr/Lr R
B	Ingress	--/L F /Lr A		
	Transit	L/L F /Lr A	Lb/Lb F /Lr A	Lr/Lr A
C	Ingress	--/L B /Lr D		
	Transit	L/L B /Lr D	Lb/Lb B /Lr D	Lr/Lr D
D	Ingress	--/L C /Lb E		

	Transit	L/L C	Lb/Lb E	Lr/Lr H
		/Lb E	/Lr H	
E	Ingress	--/L D		
		/Lb R		
	Transit	L/L D	Lb/Lb R	Lr/Lr D
		/Lb R	/Lr D	
F	Ingress	--/L G		
		/Lr B		
	Transit	L/L G	Lb/Lb G	Lr/Lr B
		/Lr B	/Lr B	
G	Ingress	--/L H		
		/Lr F		
	Transit	L/L H	Lb/Lb H	Lr/Lr F
		/Lr F	/Lr F	
I	Ingress	--/L G		
		/Lb J		
	Transit	L/L G	Lb/Lb J	Lr/Lr G
		/Lb J	/Lr G	
J	Ingress	--/L H		
		/Lr I		
	Transit	L/L H	Lb/Lb H	Lr/Lr I
		/Lr I	/Lr I	

Figure 4: Label Forwarding Entries Installed in Each Node for FEC H

Note:

1. For an ingress label forwarding entry as follows, when forward, L will be pushed and sent to the next hop A. If failure happens, Lr will be pushed and sent to the next hop E.

```
Ingress    --/L  A
            /Lr  E
```

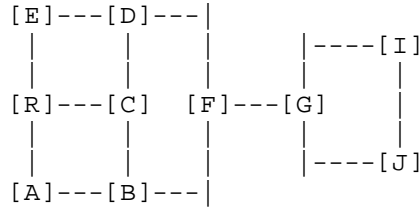
2. For a transit label forwarding entry as follows, when packet with the incoming label L arrives, L will be swapped to L and sent to the next hop A. If failure happens, L will be swapped to Lr and sent to the next hop E.

```
Transit    L/L  A
            /Lr  E
```

Above forwarding entries construct the label switch path used for fast-reroute in the forwarding plane. We can see that the existing MPLS label forwarding can be used without any extension.

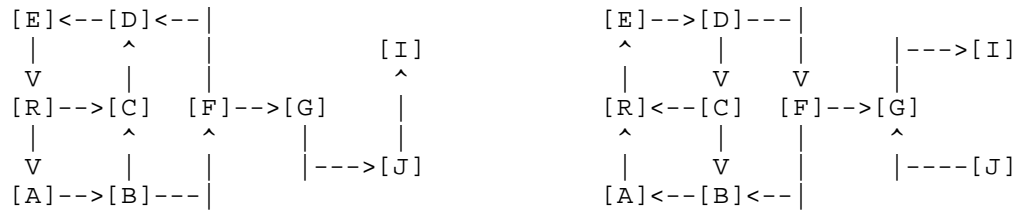
4.2. Non-2-Connected Network

[I-D.ietf-rtgwg-mrt-frr-architecture] proposes following non-2-connected network.



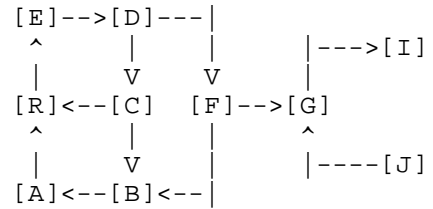
(a)

a non-2-connected graph



(b)

Blue MRT towards I



(c)

Red MRT towards I

Figure 5: A non-2-connected network

We will not explain how LDP MT is applied for the MRT FRR in detail. We choose the node I as the destination and choose R and F to observe how MRT and LDP MT work for fast-reroute.

According to MRT calculation, the path from R to I in the blue topology is R->A->B->F->G->J->I and the path from R to I in the red topology is R->E->D->F->G->I. We assume in the default topology the path from R to I is R->C->D->F->G->I. Then following forwarding entry will be created in the node R for the destination I.

		Default Topology	Blue Topology	Red Topology
R	Ingress	--/L C /Lb A		
	Transit	L/L C /Lb A	Lb/Lb A /Lr E	Lr/Lr E

Figure 6: Label Forwarding Entry of Node R for FEC I

For the node F, the path from F to I in the blue topology is F->G->J->I and the path in the red is F->G->I. We assume in the default topology the path from F to I is F->G->I. The following forwarding entry will be created in the node F for the destination I.

		Default Topology	Blue Topology	Red Topology
F	Ingress	--/L G		
	Transit	L/L G	Lb/Lb G	Lr/Lr G

Figure 7: Label Forwarding Entry of Node F for FEC I

We can see that there is no secondary route in the node F for the destination I and correspondingly there is no LDP FRR forwarding entry.

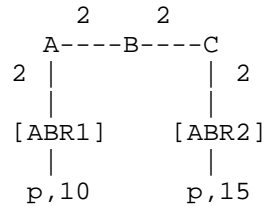
4.3. Proxy Node

There are several application scenarios proposed by [I-D.ietf-rtgwg-mrt-frr-architecture] which will use proxy node for MRT. That is, if a set of prefixes are advertised by border routers of an MRT island, a single proxy node can be used to represent the set and the proxy node and associated links are added to the network topology for MRT calculation. The application scenarios include inter-area, inter-AS and partial deployment of compatible MRT FRR routers.

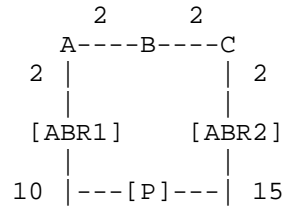
4.4. Inter-Area and Inter-AS

For Inter-area scenarios, it is desirable to go back to the default forwarding topology when leaving an area/level. There are two mechanisms proposed by [I-D.ietf-rtgwg-mrt-frr-architecture]. The first one is that ABR will advertise different labels for one specific FEC in different areas. The second one is that penultimate hop pop is done through additional computation by the penultimate router along the in-local-area MRT immediately before the ARB/LBR is reached. The first one need change of the traditional label allocation method for LDP which always distributes the same label for one FEC to all peers. When the second one used, it must be

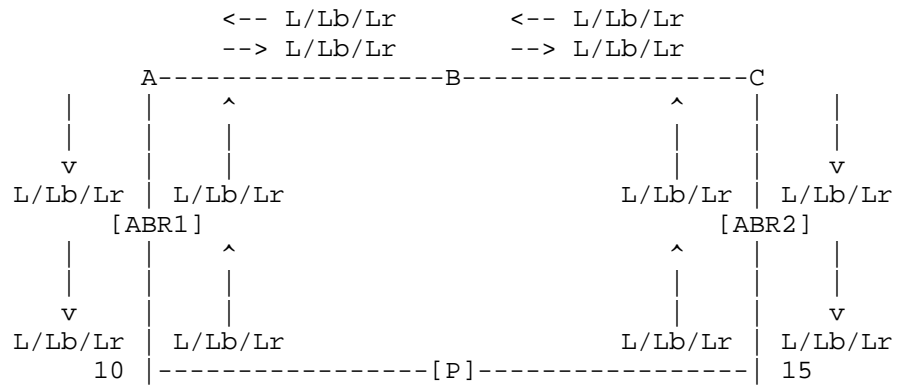
guaranteed that the IP forwarding should be done by ABR. If there is an inner label, it will cause wrong forwarding behavior. Since it is difficult to determine the type of the packet, the second mechanism must be used carefully. In order to optimize the second mechanism, when the penultimate router receive a packet with MRT label, it can swap the label to corresponding FEC's default topology label instead of penultimate hop pop.



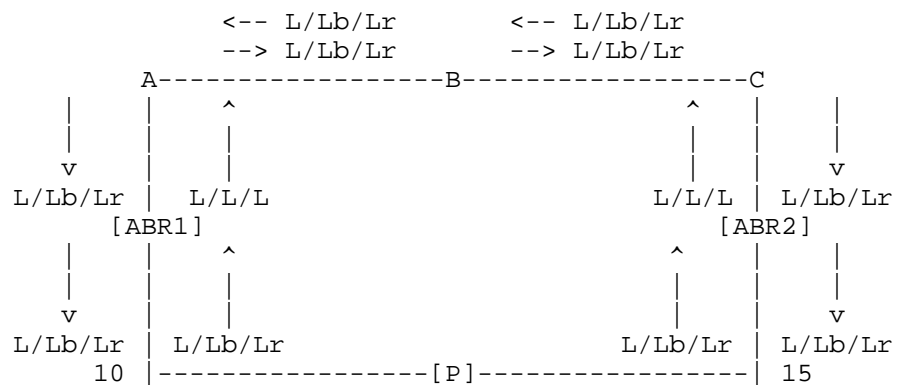
(a) Initial topology



(b) with proxy-node



(c) Label Distribution



(d) Label Distribution Change

Note: In (b),(c) and (d), label distributed by proxy nodes is actually distributed for proxy nodes by nodes in different areas from A/B/C nodes.

Figure 8: Inter-area Network and LDP MT Label Distribution for MRT FRR

According to the label distribution and MRT computation as shown in (c) of the above figure, following forwarding entries can be created in the node ABR1 and ABR2:

		Default Topology	Blue Topology	Red Topology
ABR1	Ingress	--/L P /Lr A		
	Transit	L/L P /Lr A	Lb/Lb P /Lr A	Lr/Lr A
ABR2	Ingress	--/L P /Lb C		
	Transit	L/L P /Lb C	Lb/Lb C /Lr P	Lr/Lr P

Figure 9: Label Forwarding Entry of Node ABR1 and ABR2 for Proxy Node

If the first method on change of label allocation as shown in (d) of the above figure, following forwarding entry will be created in the node A and C:

		Default Topology	Blue Topology	Red Topology
A	Ingress	--/L ABR1 /Lr B		
	Transit	L/L ABR1 /Lr B	Lb/L ABR1 /Lr B	Lr/Lr B
C	Ingress	--/L ABR2 /Lb B		
	Transit	L/L ABR2 /Lb B	Lb/Lb B /L ABR2	Lr/L ABR2

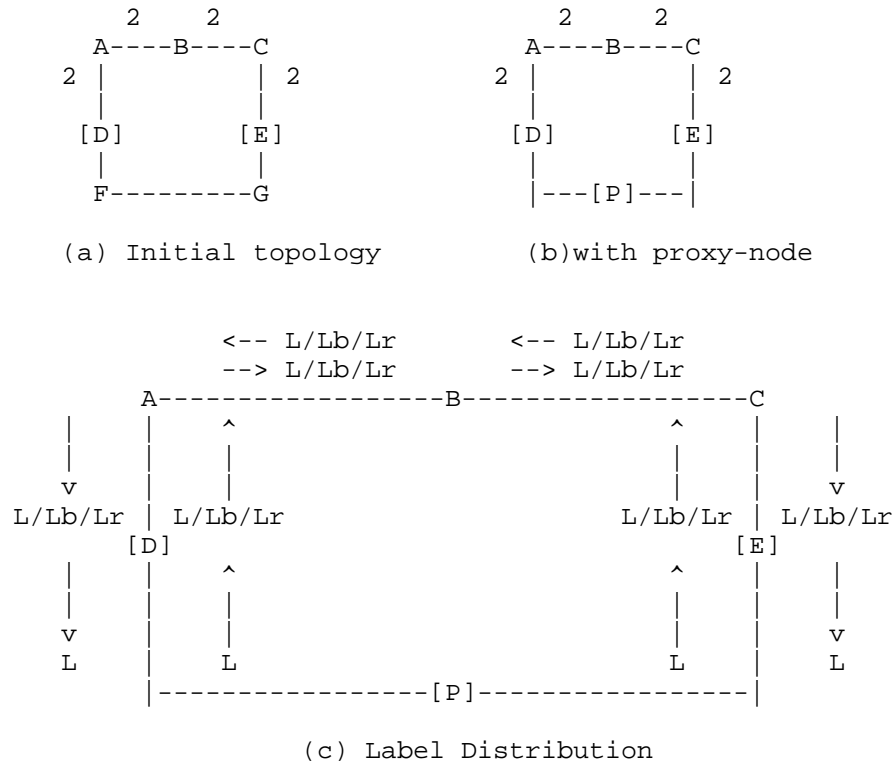
Figure 10: Label Forwarding Entry of Node A and C for Proxy Node

For inter-AS scenarios, prefixes advertised by ASBRs will set up LSP in the default topology as proxy egress. The number of prefixes will have great effect on the label allocation of LDP. When MRT fast-reroute deploys, it should be confirmed firstly that labels are enough. Or else, MRT will have negative effect on the deployment of normal service. Besides this, the complexities for ASBR protection

has been proposed by [I-D.ietf-rtgwg-mrt-frr-architecture]. It need further study.

4.5. Partial Deployment

For partial deployment and islands of compatible MRT FRR routers, proxy nodes and associated links are added to the network topology for MRT computation. The difference between partial deployment and inter-area is that in the partial deployment scenario the border nodes need proxy egress process for LDP in the blue topology and the red topology. That is, in the blue topology and red topology, the border node of the MRT network topology is not the actual egress for a prefix out of the MRT network. The border node has to advertise label for the prefix as the proxy egress.



Note: In (c), label distributed by proxy nodes is actually distributed for proxy nodes by nodes connected to [D] and [E].

Figure 11: Partial Deployment Network and LDP MT Label Distribution for MRT FRR

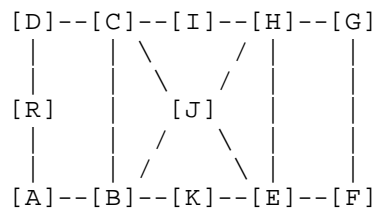
According to the label distribution and MRT computation as shown in (c) of the above figure, following forwarding entries can be created in the node D and E:

		Default Topology	Blue Topology	Red Topology
D	Ingress	--/L P		
		/Lr A		
	Transit	L/L P	Lb/L P	Lr/Lr A
E	Ingress	--/L P		
		/Lb C		
	Transit	L/L P	Lb/Lb C	Lr/L P
		/Lb C	/L P	

Figure 12: Label Forwarding Entry of Node D and E for Proxy Node

4.6. LDP over TE

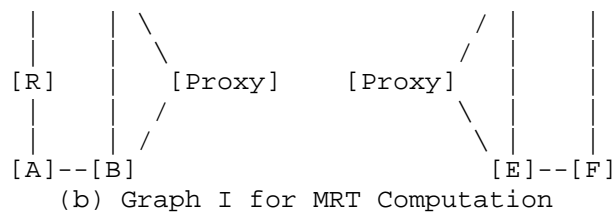
There is also additional complexity for LDP over TE scenario in which nodes in the LDP domain need to calculate two different MRT FRR for different nodes in the MPLS TE domain: edge nodes which can support both LDP and TE and internal nodes which only supports TE. Edges nodes combine with nodes in the LDP domain to form a complete topology for MRT FRR calculation. Internal nodes don't support LDP, but are not hidden from IGP topology. Some IP traffic to internal nodes which do not support LDP maybe need MRT FRR provided by nodes in the LDP domain. Nodes in the LDP domain calculate MRT FRR for these internal nodes like partial deployment. An example deployment is shown in the following figure. LDP over TE is deployed in edge nodes B, C, E and H. Internal nodes I, J and K do not support LDP. For MRT FRR, the deployment can be seen as two independent topologies. For internal nodes I, J and K, as shown in the figure (b) it is similar as the process of partial deployment. For other nodes, as shown in the figure (c) it is similar as the process of 2-connected network and the bidirectional MPLS TE paths can be used as the virtual links in MRT computation.



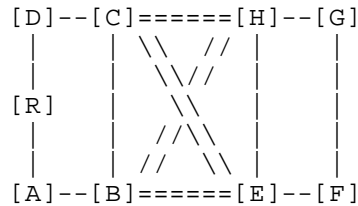
(a) Default Topology

[D]--[C]

[H]--[G]



(b) Graph I for MRT Computation



(c) Graph II for MRT Computation

Figure 13: LDP over TE Network and LDP MT Label Distribution for MRT FRR

4.7. IP-Only Network

In the IP-only network IP-in-IP has to be used. This means additional loopback addresses have to be introduced. And they are announced with associated MRT color. It will propose complexities for operation and management of the network. We recommend LDP MT should be deployed in the network for the fast-reroute usage to reduce the complexities. It also will not introduce any complexity of IP MT forwarding in the ingress node since the multi-topology only takes effect for protection. Comparing with tunnel IP packet in IP, LDP MT is an easy way to provision fast-reroute.

5. Deployment Considerations

5.1. IGP MT and LDP MT

MRT computation can be seen as a local process for IGP if only the computation is consistent for all nodes of the network. That is, multi-topology is not necessary for IGP to advertise link states with MT-ID. MT-ID is only advertised in LDP for LDP's FEC usage. That is, for MRT fast-reroute, IGP MT-ID can be independent of LDP MT-ID. But this proposes complexity for operation and management. It seems desirable to keep the consistency of MT-IDs for both IGP and LDP.

There exists another issue regarding the relationship of IGP and LDP. IGP does not support IPv4 and IPv6 in one topology. When multi-topology is used for MRT fast-reroute, the blue topology and the red topology of IPv4 should be different from those of IPv6. However,

for LDP, the address family is adopted for FEC in one multi-topology. Label distribution should be done for both IPv4 and IPv6 in one multi-topology. If the MT-ID is consistent for IGP and LDP, there should be four MT-IDs for IPv4 and IPv6 in one MRT network. For protocol extensions of MRT fast-reroute, both IPv4 and IPv6 should be taken into account for IGP to advertise information related with the blue topology and the red topology.

Besides the inconsistency of IGP MT and LDP MT, there exists the inconsistency between the OSPF MT[RFC4915] and the IS-IS MT[RFC5120]. Different MT-ID ranges for OSPF and IS-IS which cause the difficulty in reserving the same MRT MT-IDs for OSPF and IS-IS.

When multi-topology is used for MRT fast-reroute, it is error-prone for MT-ID configuration for the blue topology and the red topology on all nodes of the MRT network. In order to simplify operation and management, it is recommended that MT-IDs could be reserved for the MRT fast-reroute usage. Owing to the inconsistency of OSPF MT and IS-IS MT and the inconsistency of IGP MT and LDP MT, it seems a little challenge to reserve these possible values.

5.2. Simplified Provision

It is necessary to configure many parameters related with MRT FRR and advertise these capabilities and information by IGP[I-D.li-rtgwg-igp-ext-mrt-frr]. It is concerned that the provision complexity will have negative effect on the utility of MRT FRR.

There are two different things for the MRT FRR provision:

The first thing is the capability information which can be supported by a MRT-FRR-enabled node. The information can be directly derived without configuration.

The second thing is what parameters should be agreed on by all nodes to compute an MRT island.

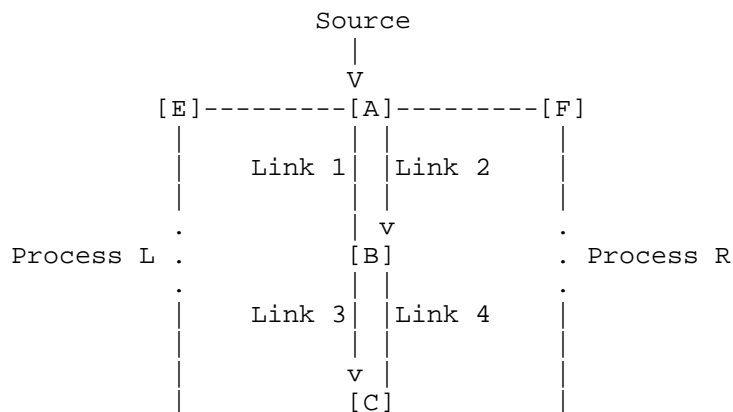
For example, as to [I-D.li-rtgwg-igp-ext-mrt-frr], a node can advertise the supported different algorithms through IGP. The supported algorithms are internal capability which is not necessary to configure. After all nodes advertise the information, they should choose one specific algorithm to compute MRT FRR. This has to be configured or all nodes should agree on a default value in advance.

The second thing should be considered more in order to simplify MRT FRR provision. In fact, LDP MT is just the method to simplify the MRT FRR provision comparing with the method that tunnels IP packet in

IP. If the latter method is preferred, it will be more difficult to design IP address carefully for each node than that only blue and red MT IDs is chosen for all nodes in the former method. There are few parameters for LDP-MT-based MRT FRR to be provisioned. The key parameters is just MT IDs and the algorithm's related parameters. As in the section 3.6, It is strongly recommended that the IGP and LDP MT IDs used for MRT FRR should be reserved. It is also the preferred default value for MRT algorithms should be defined in the appropriate documents. By this way a default profile for MRT FRR provision is determined which is composed of a set of default values. This can simplify the MRT FRR provision greatly. If it is not available, all nodes can agree on a internal default profile which is determined by the implementation and save configuration work for MRT FRR. If new nodes add to the network which use different default MT ID values and algorithm-related parameters, it can be changed administratively.

5.3. IGP Multi-process

IGP multi-process is used to isolate different areas. If an ip prefix is advertised in multiple processes, each process will calculate routes for the prefix and the shortest one will be chosen to install forwarding entry. Each IGP process calculates routes of MRT FRR independently and has its own pair of MRT topology (blue MRT topology and red MRT topology. Since the MRT paths maybe different in different process, one process' MRT next hop can not be used in another process for a specific prefix to avoid loop. So the primary route and its MRT next hop must be chosen in the same process. In order to achieve this object, there should be different blue MRT MT-IDs and red MRT MT-IDs for these processes. If there are only one pair of MRT topology for multiple processes (i.e. there is only one blue MRT MT-ID and one red MRT MT-ID), loop can happen for MRT FRR when each node chooses its primary next hop and the MRT routes in the same process for the multiple processes.



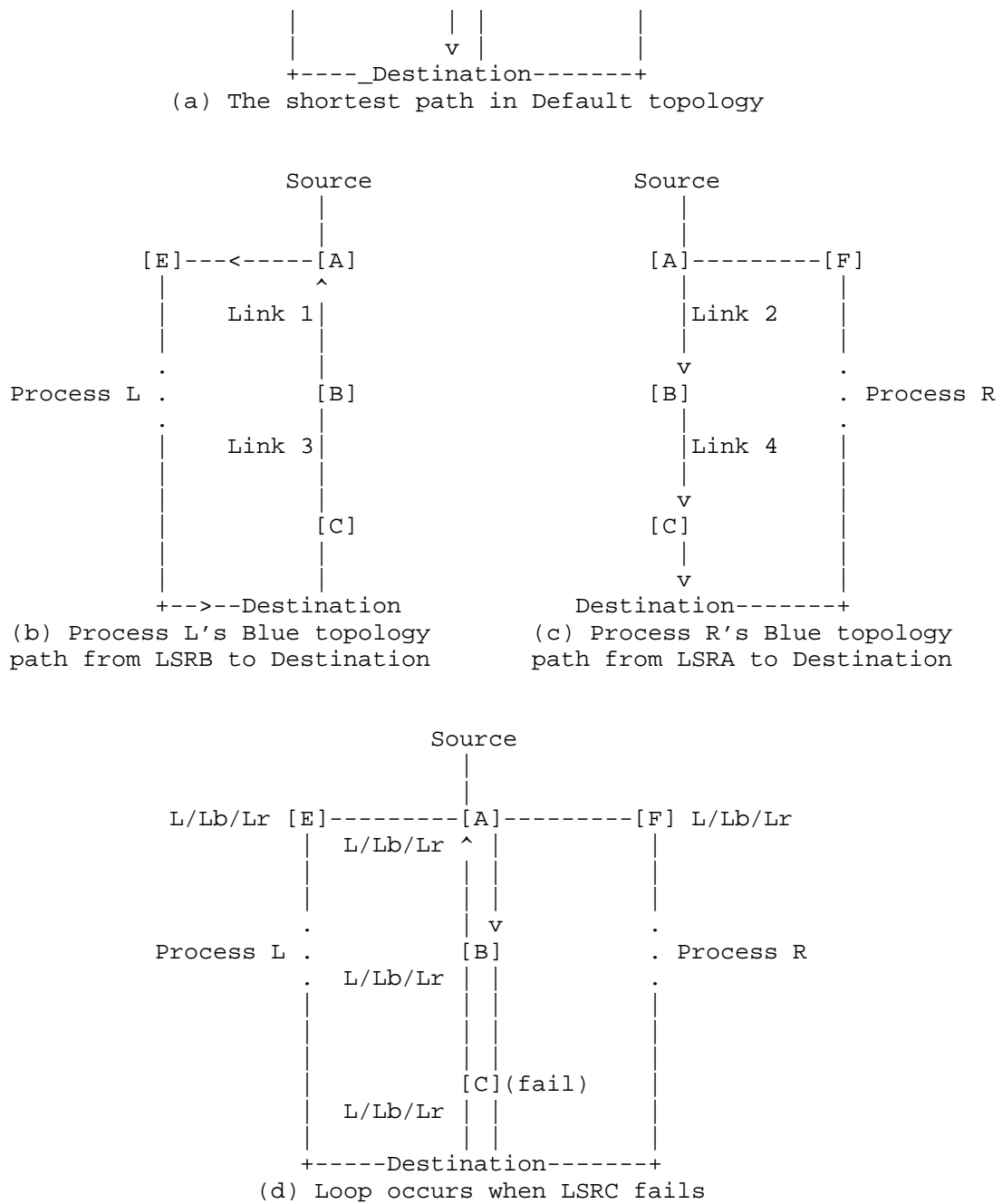


Figure 14: Loop Issue in IGP Mul-tiprocess

An IGP multi-process deployment is shown in the above figure. Node A, B and C are located in both processes: the left process L and the right process R. The process L is a ring topology, including link1 and link3. And the process R is also a ring topology, including link2 and link4. For the traffic from the Source to the Destination, assume that A chooses the shortest path determined by the process R and using link2 as the primary next hop and B chooses the shortest path determined by the process L and using link3 as the primary next hop. Process L and process R calculate MRT topologies independently, but there is only one pair of MRT MT-IDs and the label distribution is the same for different processes, this will cause the following forwarding entries are installed:

Node B: The shortest path is determined by process L. The MRT path is calculated in the same process. Assume that B calculates the blue MRT topology shown in the (b) and chooses link1 in the blue MRT topology as its secondary route. Then there is following forwarding entries for node B:

	Default Topology	Blue Topology	Red Topology
B Transit	L/L C /Lb A	Lb/Lb A /Lr C	Lr/Lr C

Node A: The shortest path is determined by process R. The MRT path is calculated in the same process. Assume that A calculates the blue MRT topology shown in the (c). Then there is following forwarding entries for node A:

	Default Topology	Blue Topology	Red Topology
A Transit	L/L B /Lr F	Lb/Lb B /Lr F	Lr/Lr F

According to above forwarding entries, if node C fails, the traffic will be sent by B to A with label Lb using the secondary route. When A receives the traffic with label Lb, it will send the traffic to B using the forwarding entry for the blue topology. Then loop happens for the traffic.

The solution of the issue is to use different MRT MTs for different processes. That is, different MRT topologies should be provisioned for different processes so that the different label distribution is done for the multiple processes. This will guarantee that when failure happens the switched traffic will be forwarded in the same process. The following figure shows the solution for as to the above example:

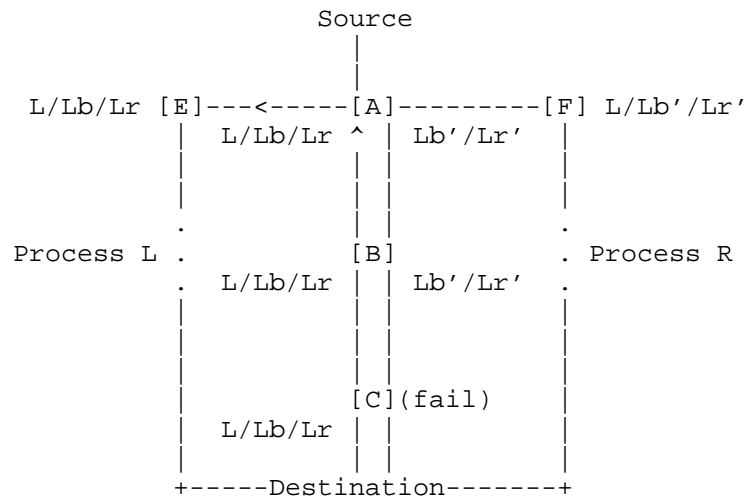


Figure 15: Separate MRT MT for Multi-process

Following forwarding entries will be created for A and B. We can see that if failure happens, the switched traffic is forwarded from A to B to E and the loop issue is avoided.

		Default Topology		Blue MT		Red MT		Blue MT'		Red MT'	
A	Transit	L/L	B	Lb/Lb	E	Lr/Lr	B	Lb'/Lb'	B	Lr'/Lr'	F
		/Lr'	F	/Lr	B			/Lr'	F		
B	Transit	L/L	C	Lb/Lb	A	Lr/Lr	C	Lb'/Lb'	C	Lr'/Lr'	A
		/Lb	A	/Lr	C			/L'	A		

Owing to the loop issue in the IGP multi-process scenario, it must be checked carefully for the reserved MT-IDs or the default profile described above for simplifying provision which will cause multiple processes share the same MRT MT-IDs. In order to prevent loop issue, separate MRT MTs for IGP multi-process have to be taken into account.

5.4. Multiple IGP

If multiple IGPs deploy in one network, the best route will be determined according to priority of these IGPs. This might cause the inconsistency issue for MRT fast-reroute. For example, when IS-IS and OSPF are deployed in one network, some nodes will use the best reroute computed by IS-IS and some nodes will use the best route computed by OSPF. If the link state is not consistent in IS-IS and OSPF, the MRT fast-reroute cannot work well. It is highly desirable that in one network only one IGP protocol is deployed and link states should be guaranteed consistent if multiple IGPs deploys.

5.5. Label Space

Advantages of LDP MT in MRT fast-reroute are apparent for simplified operation and management comparing with using IP tunnel. The main issue of LDP MT for MRT fast-reroute is resource occupancy. MRT FRR need create two redundant topologies to provide backup path. The two topologies cover all links and nodes of the MRT network. It will impact on the system resource occupancy since it will also take more resource to install routes and label forwarding entries for different topologies. When deploying LDP MT for MRT FRR, especially in the scenario of upgrading, consideration should be taken so that there is enough system resource to accommodate more routes and forwarding entries. Besides the issue related with resource occupancy, label usage is also an important issue to be taken into account. For one FEC, there are at least three label bindings distributed by one router. The number of labels for MRT fast-route is triple of that of the network without MRT fast-reroute. When LDP MT for MRT FRR is deployed, it should be guaranteed that enough labels are available so that it will not have impact on normal services such as L2VPN, L3VPN, etc.

5.6. Proxy Egress

In several scenarios where MRT FRR is deployed, proxy egress LSPs have to be setup by LDP. The proxy egress LSP maybe not end-to-end to bear VPN service in the network. But it will deteriorate label usage if LDP MT is deployed for MRT FRR. It is highly desirable that such unnecessary LSPs should be prohibited to setup to facilitate MRT FRR deployment.

5.7. Policy Control

Policy can be used to reducing the effect of more labels for MRT FRR. It is important to control on the setup of LSP in the default topology. There are two basic scenarios. The first one is the IP-only network. It is difficult to control the number of LSPs for protection since LDP MT is an extension for IP to implement protection. The second one is the multi-service network based on VPN. Policy can be applied to permit only host addresses to setup LSPs.

Policy is not recommended to control on LSP in the blue topology and the red topology since it is easy to cause inconsistency of the protection. For example, if one node need to set up MRT backup LSP for one FEC but this FEC is not allowed creating LSP by the policy in the MRT topologies, then the node cannot create the MRT backup LSP.

5.8. Resource Allocations

During the deployment of this solution, more system resource and extra label occupancy must be taken into account to avoid the possible resource exhausting.

5.9. LDP DoD

LDP DoD is used in some scenarios such as Seamless MPLS[I-D.ietf-mpls-seamless-mpls]. When MRT fast-reroute is deployed, label request will be sent according to the path calculated for different topology. The label forwarding entry will be created as the method above. Comparing with LDP DU, there are less label binding distribution for LDP DoD. In addition, LDP DoD is always used combining with conservative label retention mode. Thus there is no label binding distributed for the secondary route calculated in the default topology so that LFA cannot not be used easily. The label forwarding entry in the blue topology or the red topology will be used as the secondary one directly.

6. IANA Considerations

This document makes no request of IANA.

7. Security Considerations

There is no security issue introduced by this specification.

8. Acknowledgements

9. Normative References

[I-D.enyedi-rtgwg-mrt-frr-algorithm]

Atlas, A., Enyedi, G., Csaszar, A., and A. Gopalan,
"Algorithms for computing Maximally Redundant Trees for IP
/LDP Fast- Reroute", draft-enyedi-rtgwg-mrt-frr-
algorithm-02 (work in progress), October 2012.

[I-D.ietf-mpls-ldp-multi-topology]

Zhao, Q., Fang, L., Zhou, C., Li, L., and K. Raza, "LDP
Extensions for Multi Topology Routing", draft-ietf-mpls-
ldp-multi-topology-06 (work in progress), December 2012.

[I-D.ietf-mpls-seamless-mpls]

Leymann, N., Decraene, B., Filsfils, C., Konstantynowicz,
M., and D. Steinberg, "Seamless MPLS Architecture", draft-
ietf-mpls-seamless-mpls-02 (work in progress), October
2012.

- [I-D.ietf-rtgwg-mrt-frr-architecture]
Atlas, A., Kebler, R., Envedi, G., Csaszar, A., Tantsura, J., Konstantynowicz, M., White, R., and M. Shand, "An Architecture for IP/LDP Fast-Reroute Using Maximally Redundant Trees", draft-ietf-rtgwg-mrt-frr-architecture-02 (work in progress), February 2013.
- [I-D.li-rtgwg-igp-ext-mrt-frr]
Li, Z., Wu, N., and Q. Zhao, "Routing Extension for Fast-Reroute Using Maximally Redundant Trees", draft-li-rtgwg-igp-ext-mrt-frr-01 (work in progress), March 2013.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC4915] Psenak, P., Mirtorabi, S., Roy, A., Nguyen, L., and P. Pillay-Esnault, "Multi-Topology (MT) Routing in OSPF", RFC 4915, June 2007.
- [RFC5120] Przygienda, T., Shen, N., and N. Sheth, "M-ISIS: Multi Topology (MT) Routing in Intermediate System to Intermediate Systems (IS-ISs)", RFC 5120, February 2008.
- [RFC5714] Shand, M. and S. Bryant, "IP Fast Reroute Framework", RFC 5714, January 2010.

Authors' Addresses

Zhenbin Li
Huawei Technologies
Huawei Bld., No.156 Beiqing Rd.
Beijing 100095
China

Email: lizhenbin@huawei.com

Tao Zhou
Huawei Technologies
Huawei Bld., No.156 Beiqing Rd.
Beijing 100095
China

Email: tao.chou@huawei.com

Quintin Zhao
Huawei Technologies
125 Nagog Technology Park
Acton, MA 01719
US

Email: quintin.zhao@huawei.com

Tianle Yang
China Mobile
32, Xuanwumenxi Ave.
Beijing 01719
China

Email: yangtianle@chinamobile.com

Routing Area Working Group
Internet-Draft
Intended status: Standards Track
Expires: October 4, 2013

S. Litkowski
Orange
April 2, 2013

Node protecting remote LFA
draft-litkowski-rtgwg-node-protect-remote-lfa-00

Abstract

This draft describes a simple extension to the remote LFA specification that computes a guaranteed node protection.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on October 4, 2013.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must

include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
2. Specification	3
2.1. Reference topology	4
2.2. Current RLFA behavior	4
2.3. Guaranteed node protection computation	4
2.3.1. Computing node protecting extended P-Space	5
2.3.2. Computing node protecting Q-Space	5
2.3.3. Computing node protecting PQ-Spaces	5
3. Scaling	6
4. Security Considerations	7
5. Acknowledgements	7
6. IANA Considerations	7
7. Normative References	7
Author's Address	7

1. Introduction

The current RLFA specification described in [I-D.ietf-rtgwg-remote-lfa] is based on a per link computation (for optimization) that prevents determination of guaranteed node protection. This does not mean that the current solution is not able to provide node protection for a specific destination, but as computation is done from the link perspective, there is no guarantee to have node protection.

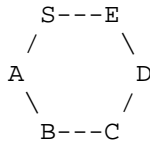


Figure 1

In the figure above, considering all metrics equal, primary path from S to D is SED. S has no LFA to reach D.

Using RLFA specification, C is a PQ from E perspective and could be used as a remote LFA in case of SE failure. In the diagram above, C is considered as a link protecting alternate (as from E perspective, it is only link protecting), even if from a topological point of view, C would be a node protecting RLFA.

There could be multiple reasons to ensure node protection in a network :

- o Presence of nodes with no high availability mechanism.
- o Avoiding transient loops in case of node crash.

This draft describes a solution to compute a guaranteed node protection using remote LFA solution.

2. Specification

2.1. Reference topology

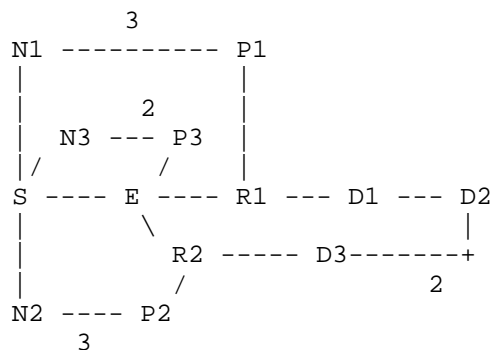


Figure 2

2.2. Current RLFA behavior

The current remote LFA computation is based on computation of extended P-Space and Q-Space and then intersect both to determine PQ nodes that would be used for traffic tunneling. As Q-Space requires rSPF computation (one per Q-Space), it would not be scalable to compute Q-Space for each prefix. RLFA specification is proposing to compute Q Space on a per link basis, PQ will so be used by all prefixes using the link as primary nexthop.

Using figure 2 and considering protection of prefixes using E as primary nexthop on S, there would be three nodes in PQ space : P1,P2,P3. Using shortest path to PQ as tie breaker, P3 would be the best PQ.

P3 is only providing link protection to D1, D2 and D3 while P1 was providing node protection for D1 and D2, and P2 was providing node protection for D3.

In this scenario, RLFA is providing a suboptimal protection because of the approximation of using remote end of the link for computation.

2.3. Guaranteed node protection computation

To compute a guaranteed node protection using remote LFA, we just introduce a small modification in the current algorithm. Rather than computing link protecting PQs from nexthop perspective, we propose to compute node protecting PQs from nextnexthop perspective.

Our proposal is working as follows :

- o Compute node protecting PQ space for each nextnexthop.
- o Compute link protecting PQ space for each nexthop (as already done).
- o Prefixes are inheriting protection type from nexthop or nextnexthop based on defined policies. Choosing a PQ in node protecting space provides guaranteed node protection.

2.3.1. Computing node protecting extended P-Space

We define the notion of node protecting extended P-Space of a node S for a connected node E as the union of :

- o Node protecting P-Space : the set of routers reachable from S without traversing node E.
- o The set of destinations using E as primary nexthop but having a node protecting LFA.

2.3.2. Computing node protecting Q-Space

The set of routers from which a node N can be reached, by normal forwarding, without traversing the node E is termed the node protecting Q-space of N with respect to the node E. The Q-space can be obtained by computing a reverse shortest path tree (rSPT) rooted at N, with the sub-tree which traverses the failed node excised (including those which are members of an ECMP).

2.3.3. Computing node protecting PQ-Spaces

Upon termination of regular SPT, node S has knowledge of nexthops and nextnexthops to reach every destinations. For each nextnexthop on the SPT, S will compute :

- o Node protecting extended P-Space considering nexthop will fail.
- o Node protecting extended Q-Space from nextnexthop considering nexthop will fail.

Intersection of both will result in node protecting PQ-Spaces and node S will have a node protecting PQ-Space for each nextnexthop.

In figure 2, considering link SE, S has two nextnexthops (R1 and R2). S will compute node protecting PQ Space for R1 and for R2.

Node protecting extended P-Space :

- o For R1 : P1,N1
- o For R2 : P2,N2

Node protecting Q-Space :

- o For R1 : P1,D1,D2
- o For R2 : P2,D3,D2

Node protecting PQ-Space :

- o For R1 : P1
- o For R2 : P2

As a result, P1 is able to provide node protection for all destinations using R1 as nextnexthop (D1,D2). P2 is able to provide node protection for all destinations using R2 as nextnexthop (D3).

3. Scaling

Approximation introduced in remote LFA specification was done to optimize computation. Our proposal is introducing more computation but in a very acceptable degree. Computation time will be dependent on number of nextnexthops rather than nexthops.

Topology	Min	Average	Med	95th perc	Max
T1	1	20,4	19	39	107
T2	1	9,5	6	27	35
T3	2	15,7	11	41	53
T4	1	13,9	15	26,5	30
T5	1	12	8	36	72
T6	2	4,3	4	7	8
T7	1	9	6	19	22

Table 1: Number of nextnexthop

The simulation results above (from real SP topologies) show that the number of rSPF to compute is really acceptable : SPF takes today only few msec to compute even on large topologies. Moreover installing protection is not an urgent task, and it would be better to take more

time to compute a more optimal protection.

4. Security Considerations

This document does not introduce any change in security consideration compared to [RFC5286] or [I-D.ietf-rtgwg-remote-lfa].

5. Acknowledgements

6. IANA Considerations

This document has no action for IANA.

7. Normative References

- [I-D.ietf-rtgwg-remote-lfa]
Bryant, S., Filss, C., Previdi, S., Shand, M., and S. Ning, "Remote LFA FRR", draft-ietf-rtgwg-remote-lfa-01 (work in progress), December 2012.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC5286] Atlas, A. and A. Zinin, "Basic Specification for IP Fast Reroute: Loop-Free Alternates", RFC 5286, September 2008.

Author's Address

Stephane Litkowski
Orange

Email: stephane.litkowski@orange.com

rtgwg
Internet-Draft
Intended status: Informational
Expires: January 16, 2014

Vic. Liu
China Mobile
July 15, 2013

Problem Statement for MTU Configuration
draft-liu-rtgwg-mtu-config-ps-00

Abstract

This document describes problem statement of MTU configuration IP and MPLS network along with the test case and result in multi-vendor network. Necessary considerations and recommendation are also discussed.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on January 16, 2014.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Terminology	2
3. Requirements	2
4. Problem statement	3
4.1. ISIS negotiations and configuration	3
4.2. MPLS MTU configuration	4
5. Test on MTU configuration	4
5.1. Basic test description	4
5.2. Test result and analysis	7
6. Considerations and recommendations	10
7. Acknowledgements	11
8. IANA Considerations	11
9. Security Considerations	11
10. Informative References	11
Author's Address	11

1. Introduction

MTU for Jumbo frames is needed in practical network, This document describes a test on MTU configuration based on IS-IS and MPLS protocol with routers from different vendors. By test of upgrading MTU from 1500 to 4470 in multi-vendor network, we bring up a problem statement of MTU configuration that measure ISIS negotiation. With the discussion, necessary considerations and recommendation are also discussed.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

2. Terminology

Throughout this document, the use of the terms "Provider Edge (PE) and Customer Edge (CE)" or "PE/CE" will be replaced by "PE" in all cases except when a network device is a CE when used in the carrier's carrier model.

3. Requirements

This section briefly describes enterprise customers and operational requirement on network MTU configuration along with a list of problems on MTU configuration over multi-vendor equipment.

The MTU default configuration is 1500 in China Mobile backbone network. However some enterprise customers requested that IP datagram's up to a length of 1600 bytes (including the IP header,

aka. baby giant) must not be fragmented in the supplier's network (from CE to CE). The receiving sequence of data packets at the destination location must match the sending sequence at the source location. Besides, from the network operational point of view, TD-LTE network also has the MTU requirement. As the CE connect to eNodeB with packet of 1500. It will be jumbo frame in S1-U port(GTP header will be added). To avoid fragment from CE to CE, it required MTU to support IP datagrams length of 1600 bytes at least.

In above, we decided to upgrade MTU from 1500 to 4470(same as POS MTU) in the backbone routers. In the rest of this draft, we are introduce a multi-vender test on MTU from 1500 to 4470 in IP networks and MPLS-VPN networks along with problem statement during this configuration.

4. Problem statement

This section list a number of problems we met during configure MTU from 1500 to 4470 in IP networks and MPLS-VPN networks with routers from different vendors.

4.1. ISIS negotiations and configuration

In IP networks, we tested routers from 4 vendors, indicated as A, B, C and D as the rest of the draft. As all the four router being configure to 4470. The ISIS protocol status is listed as table 1 below. ISIS negotiation appears 'Init' as MTU being configured to 4470 across 4 routers from different vendors.

Router 1	Router 2	ISIS status
A	B	UP
B	C	Init
C	D	UP
D	A	Init
B	D	Init
A	C	Init

Table 1 Probelm of ISIS negotiation status

4.2. MPLS MTU configuration

In MPLS networks, we figure out that the MPLS diagrams should be calculate to MTU configuration. However, as we configured and add 2 MPLS labels (8 bytes) to MTU in this 4 router. We captured fragments on some pairs among 4 routers.

5. Test on MTU configuration

This section introduction test on MTU configuration in IP networks and MPLS networks based on multi-vendor devices.

5.1. Basic test description

This test contain 3 test cases: Fist we configured MTU from 1500 to 4470 in IP network to test ISIS status on 4 routers from different routers. To reproduce this problem introduced on 3.1. Then, based on the test result, we establish MPLS network to test how MPLS MTU cause that problem in 3.2. Finally, we set up a network hybrid with IP and MPLS to test the performance of MTU configurations. Test equipment are from 4 different vendors indicated as A, B, C and D in the following parts of discussion.

TC1: Test of ISIS MTU. The basic topology of ISIS MTU testing can be depicted as follows.

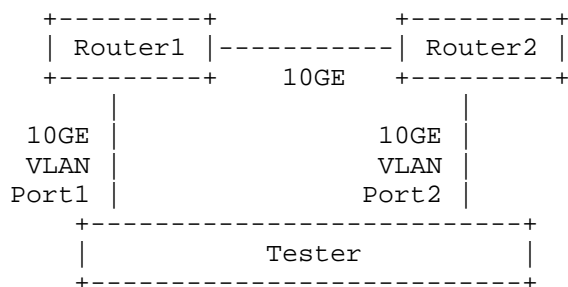


Figure 1 Basic topology for ISIS MTU testing

As figure 1 shows, two routers connect with each other by a 10GE link. Each router connect to the tester by a 10GE link. The tester generates unidirectional/bidirectional traffic between port 1 and port 2 with the traffic load of 10G. The frame length generate by tester is equals to the desired MTU parameter, 4470. We configure

MTU parameter on router 1 from 1500 to 4470 and check the ISIS protocol status and then configure the MTU parameter on router 2 from 1500 to 4470 and check ISIS protocol status again to make sure the ISIS status is up. If ISIS is not up, we increase MTU on one side until the ISIS status is up. As ISIS is established, we inject the traffic from the tester and capture the packet on port 1 and port 2 to check if there is a fragment on the packet.

The routers from 4 vendors are being tested as the sequence table below.

Router1	Router2
A	B
B	C
C	D
D	A
B	D
A	C

Table 2 TC1 test sequence table

TC2: The test topology of MPLS-VPN test can be depicted as follows.

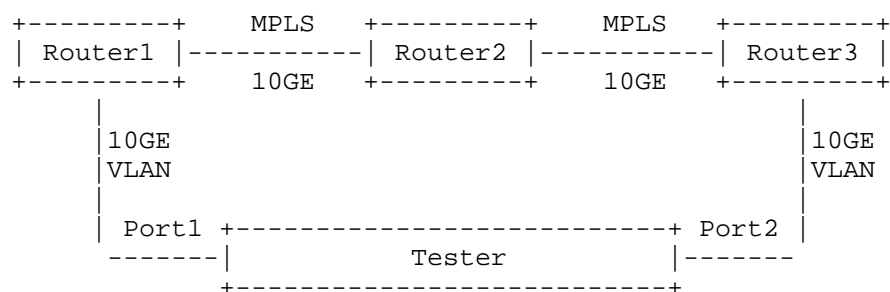


Figure 2 Basic topology for MPLS-VPN MTU testing

As figure 2 shows, there are 3 routers from 4 different vendors to build a MPLS network which Router1, Router3 take part as LER, Router 2 as LSR. VLAN are been established between PE and Tester. The whole network connect by 10GE link as the figure shows. The tester generates unidirectional/bidirectional traffic between port 1 and port 2 with the traffic load of 10G. While MTU parameter is set as the ISIS MTU test of each vendors. And the tester generate traffic with the packet length of 4474(4474+4 bit VLAN tag). We capture packet on tester's port1 and port2 to check if there is a fragment packet. If there is, we increase one of the router's MTU parameter and capture again.

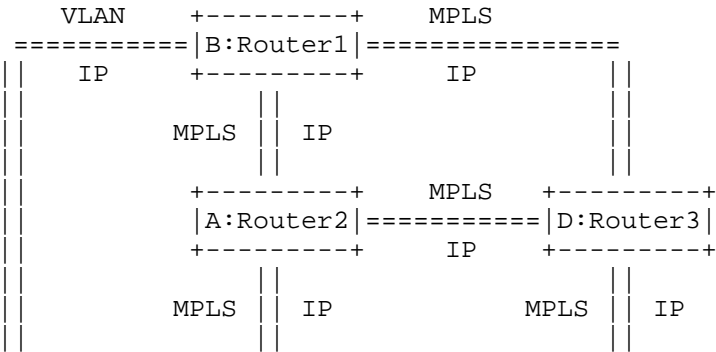
The routers from 4 vendors are being tested as the sequence table below.

Router1	Router2	Router3
B	A	C
A	B	C
B	C	D
A	D	D

Table 3 TC2 test sequence table

TC.3 MTU performance test

This test case mainly test the performance on IP and MPLS networks. The topology can be depicted as below



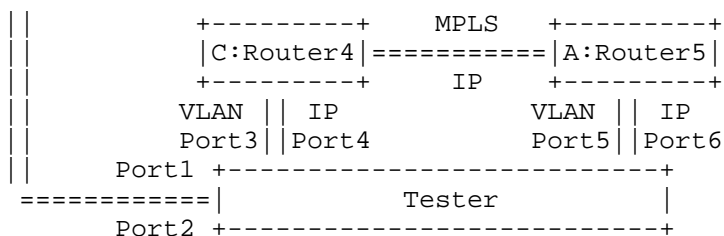


Figure 3 Basic topology for MTU performance test

As figure 3 shows, there are 5 routers to setup 4 vendor network with both MPLS and IP traffic. Router1, Router4 and Router5 act as PE while Router2 takes part as RR and Router3 as P. The whole network is being connect by 10GE links. The Tester connect with PE by two links while one is for VLAN traffic and other is for IP traffic. PE connect with RR and P also using two links. One is for MPLS traffic and other is for IP traffic. All ports on tester generate traffic load of 10GE with random frame length (minimum 64, MAX is equal to MTU of PE). The MTU parameter of routers on each port is set as the result in TC1 and TC2.

This test case is to check the performance of latency and packet dropping rate with 24 hours.

5.2. Test result and analysis

This section introduce test report of MTU configuration workable in IP network and MPLS networks.

A. Report of MTU configure in IP network based on multi-vendor routers. In test case 1, ISIS MTU test. We firstly set all configure MTU to 4470 and we get the ISIS establish status as this table below

	A	B	C	D
A	\	\	\	\
B	UP	\	\	\
C	Init	Init	\	\
D	Init	Init	UP	\

Table 4 ISIS status with equal MTU

By analyze the IP diagram figure below. We figure out the router of vendor A and vendor B is being configure based on L3 MTU which means the MTU parameter equals to L3 diagram length, 4470. And Vendor C and Vendor D is being configure based on L2 MTU which means the MTU configure parameter equals to L2 diagram length(without FCS), 4488

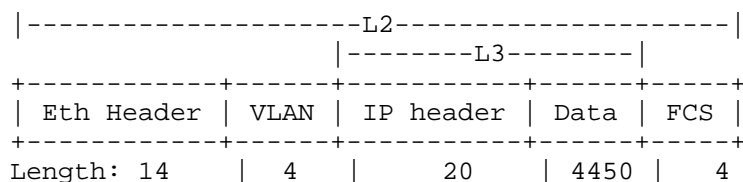


Figure 4 IP diagram

We find out the minimum MTU configuration parameter that allow ISIS to establish as the table below:

Vendor	MTU	Configure
A	4470	L3 MTU
B	4470	L3 MTU
C	4484+4(VLAN)	L2 MTU
D	4484+4(VLAN)	L2 MTU

Table 5 ISIS status with equal MTU

VLAN tag configuration in vendor A and vendor B is not counted in the MTU configuration. However, in vendor C and vendor D, because of its L2 MTU, so we have to count every tag in the IP diagram.

B. ISIS MTU negotiation test result between routers from different vendors.

As test case 1, we modified the MTU from 1500 to 4470 on one side to another to test ISIS establish status as the table blow. (With 1 VLAN tag)

	A:4470	B:4470	C:4488	D:4488
B:1500	\	Init	UP	UP
B:1500	Init	\	UP	UP
C:1518	Init	Init	\	UP
D:1518	Init	Init	UP	\

Table 6 MTU measurement of ISIS UP/Init status

By capture the packet from the ports on the tester and analysis of this result table, the negotiation method from vendor A and vendor B is strict to its MTU. However the Vendor C and vendor D is check MTU with the other side.

C. MTU influence on the MPLS with multi-vendor router. In test case 2, we test MTU configuration on multi-vendor routers in MPLS-VPN networks. The MPLS diagram can be figure as figure 5

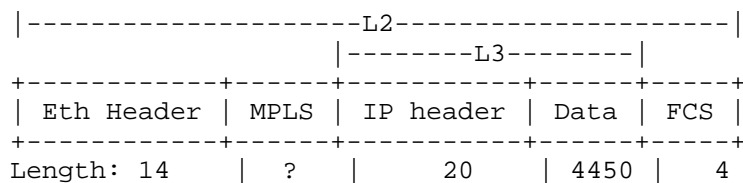


Figure 5 MPLS diagram

The ? marker can be explained as the MPLS frame length depends on how many labels used in this network. In our test models, the default MPLS labels is two(8 bits). As we configure the MTU 4470L3 on Vendor A and B (4484 L2 on Vendor C and D) on all of the fours router. We capture the packet from the tester's receiver ports and find

fragment. As we used two Vendor A's router with one Vendors B's router to build a MPLS network A-B-A and set the MTU parameter as 4470. There is no fragment. As the MTU is L2 MTU on vendor C and vendor D. We add two MPLS labels in C's and D's MTU but is still have fragment. As we increase C's and D's MTU. We get the result as:

A	4470+8	Add 2 MPLS labels
B	4470+8	Add 2 MPLS labels
C	4484+20	Default is 5 MPLS labels and cannot modify
D	4484+12	Default is 3 MPLS labels and can modify manually

Table 7 MPLS MTU configuraton of different verndors

6. Considerations and recommendations

In this section, we summarize the analysis and come to the following considerations:

a.Unified IP MTU negotiations should be taken into consideration. As we know from the test. The MTU configuration can be divided into two kinds, some vendor configure MTU based on L2 IP diagram, Vlan tag is not counted into MTU configuration. However some vendor configure MTU by L3 diagram which means MTU configuration should count in all diagram length. That will make confuse multi-vender in networking operation.

b.Unified MPLS MTU configuration. In MPLS-VPN network, MPLS label should be calculate in MTU configuration. However, in multi-vendors networks, there is no standardization to make a unified MPLS MTU configuration. Vendors such as A and B, It add MPLS labels manually as configured. Vendor C add 5 fixed labels. Vendor D add 3 fixed label. If the MPLS label is not unified in MTU configuration, it will cause fragment in network operation.

c.Unified default MTU negotiation. Vendor A and Vendor B is compare MTU by a strict mode in MTU negotiation that MTU in both side should be equal to establish ISIS. However Vendor C and Vendor D compare

MTU by a loose mode in MTU negotiation that ISIS is established by Minimum MTU of one side. Although it easier to for MTU negotiation, it's much easier to cause fragments for multi-vendor networking operations.

With the consideration above, we summarize recommendations follows:

We test MTU configuration in IP network and MPLS network with routers from different vendors. The variety forms of MTU configuration trigger fragment easily in multi-vender networks. However, in order to make network operation easier, we hope to bring out this problem statement for all vender to standardize the MTU configuration in IP network and MPLS network. Meantime, we want to write an operation guideline to open the test result to declare the problem and guide others in MTU configurations.

7. Acknowledgements

8. IANA Considerations

The IANA has assigned { mplsStdMIB 11 } to the MPLS-L3VPN-STD-MIB module specified in [RFC 4362]. This document only makes extensions to the MPLS-L3VPN-STD-MIB module, there is no further IANA requirement.

9. Security Considerations

No specific security issues with the proposed solutions are known. The proposed extension in this document does not introduce any new security considerations beyond that already apply to the base MPLS/BGP L3VPN specification as [RFC 3031] and [RFC 4364].

10. Informative References

- [RFC1981] McCann, J., "Path MTU Discovery for IP version 6", RFC 1981, August 1996.
- [RFC4821] Mathis, M., "Packetization Layer Path MTU Discovery", RFC 4821, March 2007.

Author's Address

Vic Liu
China Mobile

Email: LiuZhiheng@chinamobile.com

Routing Working Group
Internet-Draft
Intended status: Standards Track
Expires: July 28, 2014

IJ. Wijnands, Ed.
L. De Ghein
Cisco
G. Enyedi, Ed.
A. Csaszar
J. Tantsura
Ericsson
January 24, 2014

Tree Notification to Improve Multicast Fast Reroute
draft-wijnands-rtgwg-mcast-frr-tn-02

Abstract

This draft proposes dataplane triggered Tree Notifications to support multicast fast reroute for PIM and mLDP. These Tree Notifications are initiated by a node detecting the failure to a Repair Node downstream. A Repair Node is a node that has a pre-built backup path that can circumvent the failure. Using this mechanism, a Repair Node has the ability to learn about non-local failures quickly without having to wait for the IGP to convergence. This draft also covers an optional method to avoid bandwidth usage on the pre-built backup path.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on July 28, 2014.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal

Provisions Relating to IETF Documents
(<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Terminology and Definitions	3
2. Introduction	4
3. Improving Non-local failures	4
3.1. Downstream Tree Notifications	5
3.2. DTN processing logic	5
3.3. Repair Node discovery	7
3.3.1. Repair Node Information item	8
4. Reduce the bandwidth consumption in networks with fast failover response times	8
4.1. Joining a secondary tree in blocking mode	9
4.2. Upstream Tree Notifications	9
5. MRT/MCI-Only Mode	10
6. TN Authentication	10
7. The TN Packet	11
7.1. TN Packet Format	11
7.1.1. TN TimeStamp TLV Format	13
7.1.2. TN Signature TLV Format	13
8. PIM Specific TN Components	14
8.1. RNI item in PIM Join Message	14
8.2. Tree Information Item	16
8.3. Incremental deployment	17
9. mLDLP Specific TN Components	17
9.1. RNI item in mLDLP Label Mapping	18
9.2. Tree Information Item	19
10. Acknowledgements	19
11. IANA Considerations	20
12. Security Considerations	20
13. References	20
13.1. Normative References	20
13.2. Informative References	21
Authors' Addresses	21

1. Terminology and Definitions

MoFRR : Multicast only Fast Re-Route.

LFA : Loop Free Alternate.

mLDP : Multi-point Label Distribution Protocol.

PIM : Protocol Independent Multicast.

UMH : Upstream Multicast Hop, a candidate next-hop that can be used to reach the root of the tree.

tree : Either a PIM (S,G)/(*,G) tree or a mLDP P2MP or MP2MP LSP.

OIF : Outgoing InterFace, an interface used to forward multicast packets down the tree towards the receivers. Either a PIM (S,G)/(*,G) tree or a mLDP P2MP or MP2MP LSP.

IIF : Incoming InterFace, an interface where multicast traffic is received by a router.

MCE : MultiCast Egress, the last node where the multicast stream exits the current transport technology (MPLS-mLDP or IP-PIM) domain or administrative domain. This maybe the router attached to a multicast receiver.

MCI : MultiCast Ingress, the node where the multicast stream enters the current transport technology (MPLS-mLDP or IP-PIM) domain. This maybe the router attached to the multicast source.

DTN : Downstream Tree Notification.

UTN : Upstream Tree Notification.

TN : Tree Notification, Upstream or Downstream

JM : Join Message, the message used to join to a multicast tree, i.e. to build up the tree. In PIM, this is a JOIN message, while in mLDP this corresponds to a Label Mapping message.

MRT : Maximally Redundant Trees.

Repair Node : The node performing a dual-join to the tree through two different UMHs. Sometimes also called as dual-joining node or merging node (it merges the secondary and primary tree).

RNI : The Repair Node Information is an item included in the TN which holds the necessary repair information when the TN is sent to the Repair Node.

Branching Node : A node, (i) which is considered as being on the primary tree by its immediate UMH and (ii) which has at least one OIF on the secondary tree installed for a multicast tree.

2. Introduction

Both [I-D.karan-mofrr] and [I-D.atlas-rtgwg-mrt-mc-arch] describe "live-live" multicast protection, where a node joins a tree via different candidate upstream multicast hops (UMH). With MoFRR the list of candidate UMHS can come from either ECMP or Loop Free Alternate (LFA) paths towards the MultiCast Ingress node (MCI). With MRT, the candidate UMHS are determined by looking up the MCI in two different (Red and Blue) topologies. In either case, the multicast traffic is simultaneously received over different paths/topologies for the same tree. The node 'dual-joining' the tree needs a mechanism to prevent duplicate packets being forwarded to the end user. For that reason a node 'dual-joining' the tree only accepts packets from one of the UMHS at the time. Which UMH is preferred is a local decision that can be based on IGP reachability, link status, BFD, traffic flow monitoring, etc...

Should the node detect a local failure on the primary UMH, the node has an instantly available secondary UMH that it can switch to, simply by unblocking the secondary UMH. The dual-joining node is also called Repair Node in the following.

This draft attempts to improve these solutions by:

- o Improving fail-over time and the reliability of failure detection for non-local failures; and
- o Reducing the bandwidth consumption in a network with fast failover response times, by avoiding sending the multicast traffic over the secondary path.

3. Improving Non-local failures

If a failure is not local and happens further upstream, the dual-joining node needs a fast mechanism (i) to detect the upstream failure and (ii) to learn that other upstream nodes cannot circumvent the failure. Existing methods based on traffic monitoring are limited in scope and work best with a steady state packet flow.

Therefore, we propose a method which can trigger the unblocking the secondary UMH independently of the packet flow.

Figure 1 shows an example. Consider that, e.g., node A goes down. Nodes C, D and E cannot detect that locally, so they need to resort to other means. After detecting the failure, node C should not change to its secondary UMH (node J) as it won't help for the failure of A. Node D, on the other hand, will have to unblock its secondary UMH (node I). Yet again, with MoFRR, node E should not unblock its secondary UMH (node K): (i) this won't help in resolving the failure of node A, and (ii) one of its upstream nodes (node D in this case) will be able to restore the stream with a fail-over action.

3.1. Downstream Tree Notifications

When a node detects a local failure of its primary UMH it MUST originate a Downstream Tree Notification (DTN) to all the Repair Nodes directly below it in the multicast tree. The method of discovering such nodes is described in Section 3.3. When a Repair Node receives a DTN containing the primary UMH of the node, it must switch to the secondary UMH.

DTN packets are sent to the Repair Node via unicast. The packet may be forwarded using any transport that is available (MPLS or IP) to reach the destination. The IP precedence in the IP header should have a value of 6 (Internetwork Control). The EXP field (Traffic Class field) in the MPLS header should have a value of 6. The DTN packets are identified by a well known port number (to be allocated). Using a well-known port number it is easy for the Repair Node to identify the DTN packet and invoke the procedures as described in this draft. We are proposing to allocate different port numbers for PIM and mDLP since it will be easier to dispatch the packet to the right process dealing with this request.

When a router detects a local failure, it should sent out the DTN packet to the Repair Node as fast as possible. The sooner the Repair Node gets the packet, the sooner the traffic can be restored. It is recommended that the DTN packet is pre-created and originated from the data-plane. The same is true for receiving the DTN packet on the Repair Node, the faster it can be processed, the faster the traffic is restored. For both forwarding and processing the DTN, control-plane interaction SHOULD be avoided to get the best failover results.

3.2. DTN processing logic

When a DTN packet is received on the Repair Node it must determine which tree and UMH the notification is for. The information encoded in the DTN is specific for the type of tree being used, i.e. PIM vs

mLDP. For details on the specific encoding see Section 8 and Section 9 for the details. Once the Repair node has determined the tree and the UMH, the following rules are use for processing the DTN.

1. If the UMH encoded in the DTN packet is the primary UMH in the tree, the secondary UMH MUST become the new primary UMH and the old primary MUST become the secondary.
2. If the UMH encoded in the DTN packet is the secondary UMH in the tree, no action needs to be taken.
3. If a DTN notification has been received for both the primary and secondary UMH in the tree, a new DTN notification MUST be originated to the Repair Node(s) downstream from this node.

In order for the Repair Node to determine that a DTN notification was received for both the primary and secondary UMH, it must store the fact a DTN was received for a particular UMH.

Consider the example in Figure 1 below. MCI is the root of a tree that includes the nodes as follows (based on the primary UMH).

```

->F->G->H->I
MCI
->A->B->C->D->E

```

Node C, D and E are candidate Repair Nodes.

```

-- Primary UMH
++ Secondary UMH

```

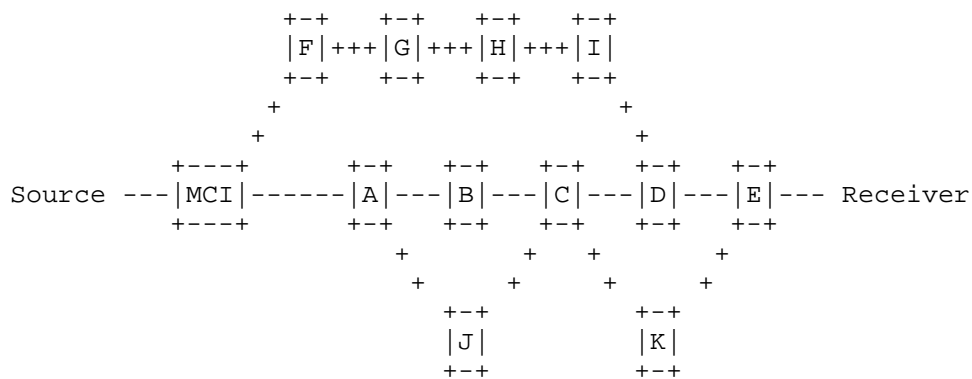


Figure 1: Remote failure example

Suppose that the link between node A and B failed, B is directly connected and will detect the failure locally. In this case, node B is the only node that detects the failure and will originate a DTN to its downstream repair node C. Node C will receive the DTN for the UMH that is the primary UMH. Following rule 1 (Section 3.2), node C will make the backup UMH the new primary. No further action is needed because C has repaired the tree via node J. Note, J would not have sent a DTN to node C because J is not directly connected to the failing link.

Suppose that node A fails, B and J are directly connected and detect the failure locally. A DTN packet is triggered to first downstream repair node of A, which is node C. Node C is an unusable Repair Node because it will receive DTN for both the primary UMH (from B) and the secondary UMH (from J). Following rule 3 (Section 3.2), C can't repair the tree and must send a new DTN packet towards the Repair Nodes of C, which are D, on the primary path, and E, on the secondary path.

Suppose that the link between A and the MCI failed. Node A is directly connected to the failure and will trigger a DTN packet to its downstream repair node(s). In this case, node A has learned about the downstream repair node C twice, the primary UMH (via node B) and secondary UMH (via node J). Node A will therefore send a DTN packet including both the primary and secondary UMH to node C (see Section 7 for details on the encoding). Following rule 3 (Section 3.2), C can't repair the tree and must send a new DTN packet towards the Repair Nodes of C, which are D, on the primary path, and E, on the secondary path.

The DTN packet that D received from C will match against the primary UMH. Following rule 1, D will activate the backup path to I. The DTN packet that E received from C will match against the backup UMH, following rule 2, no action is taken. In the example one can see that we recovered from the failure because node D started accepting the data packets from node I and is forwarding them to node E.

3.3. Repair Node discovery

In example Figure 1 we wrote that nodes C, D and E are the repair nodes. How does a node determine that it is a Repair Node? The rule is straightforward, a node that is enabled to join two UMH's, one in active the other in backup ([I-D.karan-mofrr]), is a repair node on the tree. A Repair node has the ability to repair the tree for the nodes upstream from this node. In order for the Repair Node to get notified of upstream failures (ie DTN), the nodes upstream from the Repair Node need to learn about it.

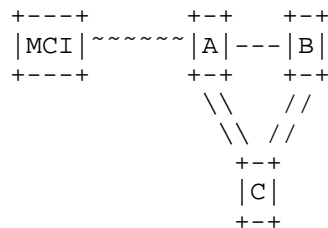
3.3.1. Repair Node Information item

A Repair Node MUST advertise its own address (either a router ID or any directly connected address) and an UMH identifier to the nodes upstream on the tree. This address and UMH are part of the RNI (Repair Node Information) item that is included in the JM. The RNI is carried hop by hop in the JM upstream. If a node along the path is not a Repair Node, it will save the RNI and forward it further upstream. If the node is a Repair Node, it will save the RNI and include its own RNI in the JM sent further upstream. If a Repair Node changes one of its UMH's, it needs to trigger a new RNI to its upstream node(s) to notify them of the changed UMH. If a RNI is received and it does not match the saved RNI, the new RNI overrides the old RNI and triggers a JM with the new RNI to its upstream node(s). A RNI includes protocol specific information on how to identify the tree and UMH. For that reason it is documented in the protocol specific sections Section 8 and Section 9.

The Repair Node MAY include additional information in the RNI for reasons of security and robustness, please see Section 6 and Section 7.1.

4. Reduce the bandwidth consumption in networks with fast failover response times

In some of networks, such as aggregation networks, bandwidth is more sparse than, e.g., in core networks. Live-live multicast protection results in more bandwidth consumption in the network as it continuously pulls traffic on both trees. In such networks it is relevant if the capacity serving backup purposes can be used, most of the time, by best-effort or even by lower-than-best-effort traffic.



Nodes A and B have receivers. Double lines show bandwidth consumption that is superfluous when there is no failure in the network.

Figure 2: Example for secondary segments occupying bandwidth in MoFRR

In live-standby mode the aim is that the secondary tree is not forwarding multicast traffic as long as there is no failure. In order to achieve such a "live-standby" multicast protection the following procedures must be followed:

- o Upstream nodes block their OIF when they are part of a standby tree.
- o If all of the OIF's of the node are marked as blocking, the node joins the tree in blocking mode further upstream.
- o A procedure so that the upstream node can quickly unblock its OIF and starts to forward.

4.1. Joining a secondary tree in blocking mode

The JM sent to the secondary UMH includes an identifier to indicate the upstream node MUST not forward packets down this branch of the tree. The identifier is TBD. The mechanism to join a secondary path is identical to what the MRT and MoFRR drafts describe, i.e. a Repair Node simply sends a secondary JM through another UMH (on another topology, in case of MRT). If a node receives a JM without a blocking identifier for an OIF that previously was in blocking mode, the blocking mode is reset and the node starts forwarding out of this interface. If this node joined the tree in blocking mode further upstream, a new JM MUST be originated to reset the blocking state further upstream.

4.2. Upstream Tree Notifications

In order to make an upstream node start forwarding on the backup path quickly after a failure was detected on the primary UMH, we send an Upstream Tree Notification (UTN) to the upstream node on the backup UMH. The failure on the primary UMH may be local or detected using a DTN. The UTN received by the upstream node should be processed in the data-plane and reset the blocking state of the OIF. If this node also joined the tree in blocking mode upstream, a UTN has to be forwarded further upstream. This procedure is repeated until we find a node that is not in blocking mode or we reached the MCI.

When the upstream node resets the blocking mode in the data-plane, the control plane will still have the blocking mode set. In order for the control plane to get in sync with the data-plane, the node that originated the UTN MUST also trigger a JM without blocking mode.

The upstream node receiving the UTN must be able to identify the tree which the notification is sent for, as well as the downstream interface it applies to. The information is encoded in a same RNI

item that is used for DTN packets. For details please see the protocol specific sections Section 8 and Section 9.

Like DTN packets, UTN packets are sent via unicast to the upstream node.

5. MRT/MCI-Only Mode

If each node in the network supports UTN and also all nodes support MRT, the nodes may work in "MRT/MCI-only" mode.

In MRT/MCI-only mode, there is one single Repair Node for all failures, the MCI. Other nodes MUST NOT consider themselves as Repair Nodes. MRT ensures the necessary maximally disjoint secondary tree up to the MCI, on a second topology. Only the MCI MUST keep its OIFs corresponding to the secondary tree blocked. Similarly, only MCEs MUST keep their secondary backup IIFs blocked. Any other nodes MUST NOT block their (secondary) IIFs or OIFs.

In MRT/MCI-only mode, the UTNP MUST be forwarded directly to the MCI. This mode enables that a node detecting a downstream failure of the primary tree MAY send a UTNP upstream towards the source/MCI on the primary tree.

If an UTNP is received by the MCI on the secondary topology in "MRT/MCI-only" mode, the MCI MUST unblock the OIF where the UTNP was received. This activates a whole sub-tree of the secondary tree.

If an UTNP is received by the MCI on the primary topology in "MRT/MCI-only" mode, the MCI gets no information on which leg to activate on the secondary tree, so it MUST activate (unblock) all secondary legs.

6. TN Authentication

If a malicious attacker can reproduce the TN packet format, unwanted reconvergence can be triggered. In order to avoid such attack, a TN packet MAY contain a digital signature. Having authentication is optional, it can be enabled or disabled in the network. If however security is enabled, all the nodes must share the same secret key, which they get either by configuration or from the multicast routing protocol. Moreover, for protection against reply attacks, each TN packet must contain a sequence number.

The sequence numbers in the network are not necessarily synchronised, instead, each node can have its own. Sequence numbers can be

generated arbitrarily, it can be even some random value; the only requirement is to create a new sequence number each time a reconvergence was triggered due to a TN (i.e. the sequence number was used).

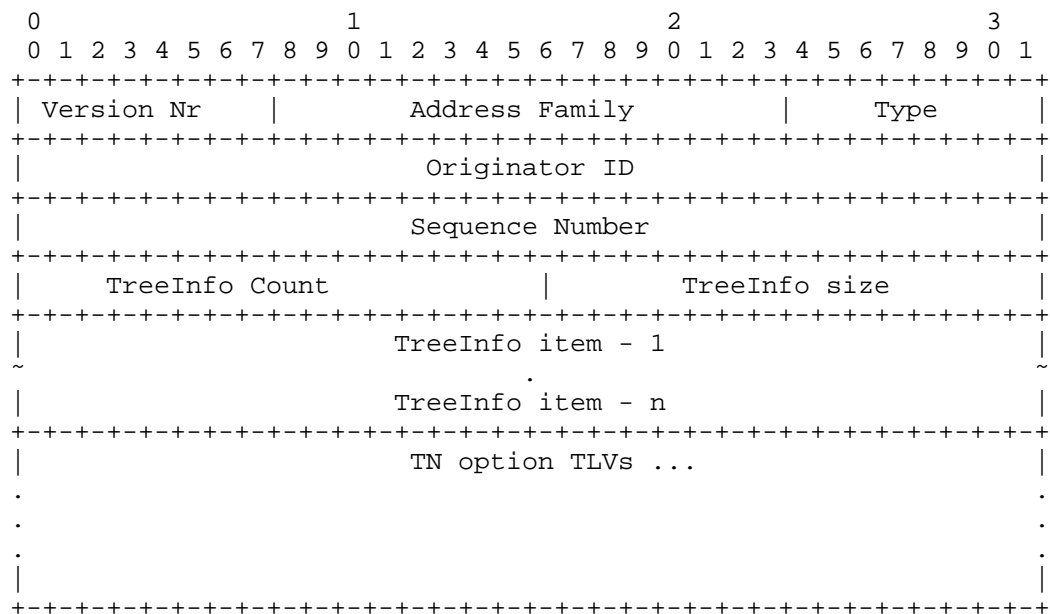
The originator of the DTN packet MUST use the sequence number of the Repair Node to create a TN signature TLV (see Section 7.1.2). For UTN packet the sender MUST use its own sequence number, what it sent previously to its UMH. The destination in this case must check validity based on the sequence number of the sender.

A sequence number is learned from JM and part of the RNI. It is the responsibility of multicast routing protocol to protect JM against malicious change.

7. The TN Packet

7.1. TN Packet Format

A Tree Notification is an IPv4 or IPv6 UDP packet with the following format.



Version number: This is a 1 octet field encoding the version number, currently 0.

Address Family: This is a 2 octet field encoding a value from ADDRESS FAMILY NUMBERS in [RFC3232] that encodes the address family for the Root Address of the tree.

Type: This is a 1 octet field encoding the message type, currently two are defined;

Type 0: Downstream Tree Notification.

Type 1: Upstream Tree Notification.

Originator ID: 4 bytes long unique ID of the originator. That can be some loopback IPv4 address if there is such, or can be set by the operator.

Sequence Number: Number unique for each failure case. It is recommend to start at 0, and to be increased by 1 each time a new TN is originated. The Sequence number may differ at each node, thus the sender and the receiver must know the same sequence number.

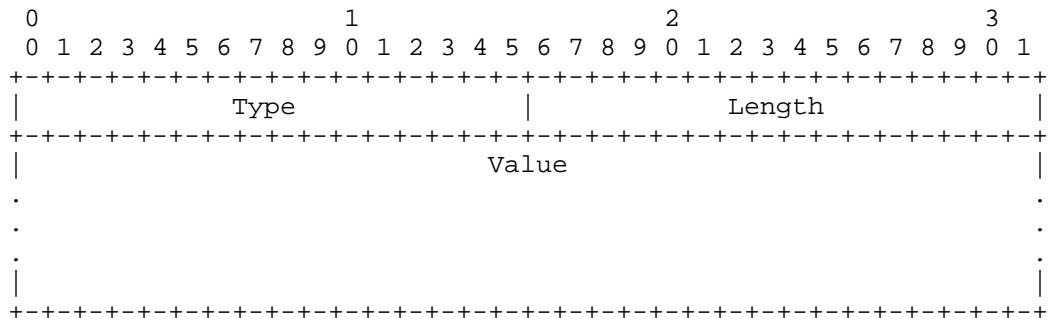
TreeInfo count: 2 octet field encoding the number of TreeInfo items includes.

TreeInfo size: 2 octet field encoding the number of octets use to encode the TreeInfo's following.

TreeInfo item: The encoding of this field is protocol specific, see Section 8 and Section 9.

TN option TLVs: TLVs (Type-Length-Value tuples) describing additional options for TN packets.

The TLV's have the following format.



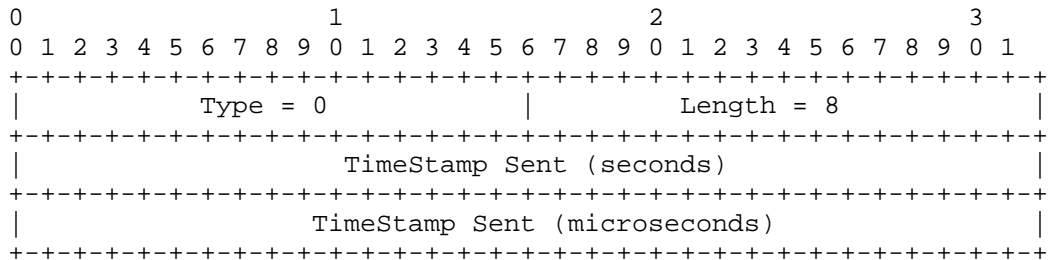
Type: This is a 2 octet field encoding the type number of the TLV.

Length: This is a 2 octet field encoding the length of the Value in octets.

Value: String of Length octets, to be interpreted as specified by the Type field.

7.1.1.1. TN TimeStamp TLV Format

The TimeStamp is an optional TLV that MAY be included when the TN was originated, it has the following format.



TimeStamp: The TimeStamp is the time-of-day (in seconds and microseconds, according to the sender's clock) in NTP format [NTP] when the Tree Notification is sent.

7.1.1.2. TN Signature TLV Format

TN Signature is an optional TLV, which protects the whole TNP (including other TLVs) against attacks thus it must be the last TLV if present. The signature is SHA-512 hash value. The input of the hash function is as follows:

```

+-----+
| Complete packet content without signature TLV |
+-----+
|                               Secret key       |
+-----+

```

Signature input: The input of the hash function is the packet extended with TN security key

The build up of the TLV is as follows:

```

0                               1                               2                               3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
|                               Type = 1                               |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                               Hash function result                     |
|                               .                                         |
|                               .                                         |
|                               .                                         |
+-----+-----+-----+-----+-----+-----+-----+-----+

```

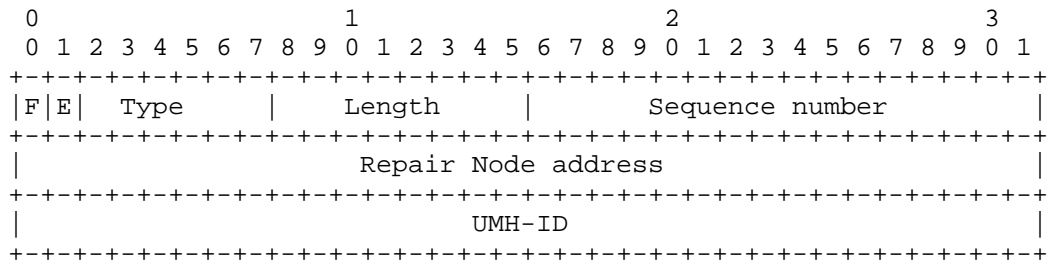
Signature: SHA-512 signature protecting TN packets.

8. PIM Specific TN Components

In this section we are documenting the PIM specific data-structures and procedures (if they are different from the generic procedures are defined in this document). As described in this document, TN packets are UDP/IP packets sent via unicast to its destination. The UDP port number for PIM is set to the (to be) assigned IANA port number for PIM-TN.

8.1. RNI item in PIM Join Message

As described previously, PIM must insert the RNI when sending a PIM join to its UMH. The RNI includes its router ID, sequence number and UMH Identifier. The UMH-ID can be locally unique identifier since its has only local significance on the Repair Node. A good ID to use would be the IP address of the interface associated with the UMH the PIM join is sent to. The RNI is carried in the PIM Join as a new PIM Attribute following [RFC5384]. The PIM RNI attribute has the following format for IPv4.



F Forward if not understood.

E End of Attributes, following [RFC5384].

Type: This 6 bit field should be assigned by IANA for TN specific JOIN messages.

Length: Length = 10 octets.

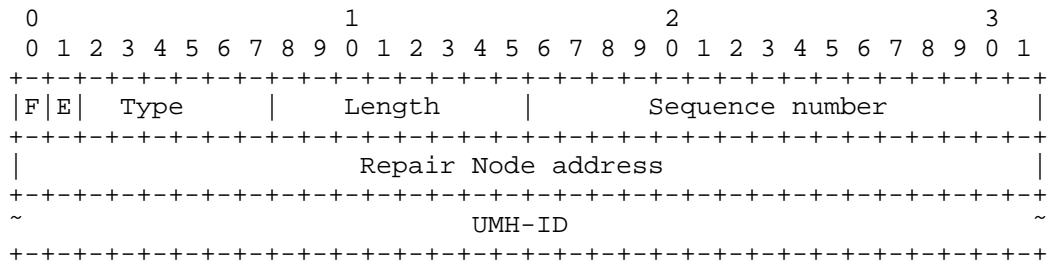
Sequence number: 2 octets long field, describing the sequence number of the sending Repair Node.

Repair Node address: The router ID of the Repair Node, in IPv4 address format.

UMH-ID: This is a 4 octet field encoding UMH identifier. This is the IPv4 address of the interface associated with the UMH the PIM join is sent to.

Figure 3: PIM IPv4 RNI attribute TLV

The PIM RNI attribute has the following format for IPv6.



F Forward if not understood.

E End of Attributes, following [RFC5384].

Type: This 6 bit field should be assigned by IANA for TN specific JOIN messages.

Length: Length = 16 octets.

Sequence number: 2 octets long field, describing the sequence number of the sending Repair Node.

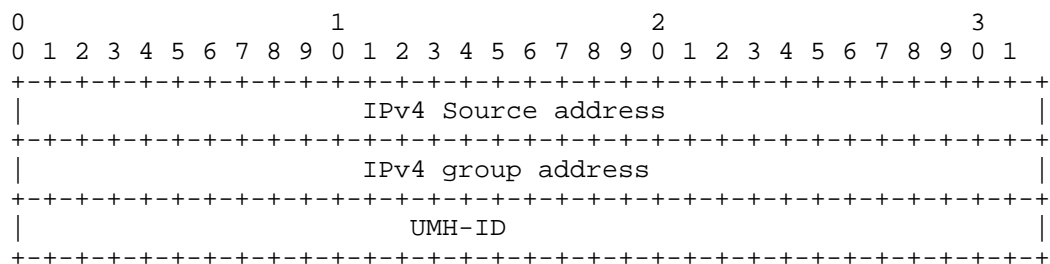
Repair Node address: The router ID of the Repair Node, in IPv4 address format.

UMH-ID: This is a 16 octet field encoding UMH identifier. This is the IPv6 address of the interface associated with the UMH the PIM join is sent to.

Figure 4: PIM IPv6 RNI attribute TLV

8.2. Tree Information Item

A TN packet contains one or more TreeInfo items that allows a Merge Node to identify which tree(s) and interface(s) are effected by the TN. The same encoding is used for DTN and UTN packets. The PIM TreeInfo items are defined for IPv4 and IPv6. Which version is to be included in the TN packet depends on Address Family in the TN packet. The UMH-ID included in the DTN MUST be taken from the RNI that was signalled for that tree. The UMH-ID for UTN packets is the PIM neighbor address for that tree. The TreeInfo item has the following format:

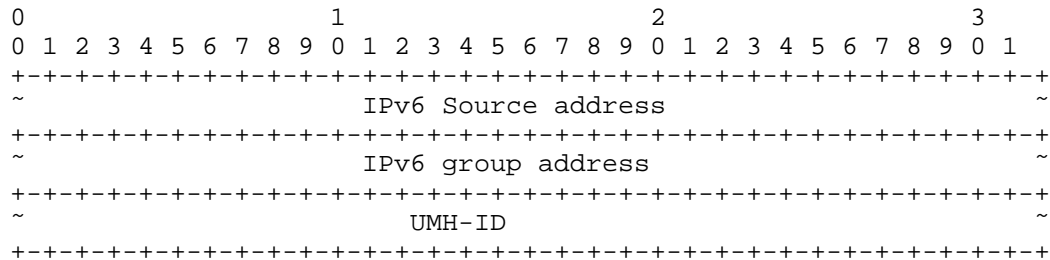


Source Address: This is a 4 octet field encoding the IPv4 source address of the tree. A source address of 0.0.0.0 means that this TN relates to a (*,G) tree.

Group Address: This is a 4 octet field encoding the IPv4 group address of the tree.

UMH-ID: This is a 4 octet field encoding UMH identifier.

Figure 5: PIM IPv4 TreeInfo item



Source Address: This is a 16 octet field encoding the IPv6 source address of the tree. A source address of 0:0:0:0:0:0:0:0 means that this TN relates to a (*,G) tree.

Group Address: This is a 16 octet field encoding the IPv6 group address of the tree.

UMH-ID: This is a 16 octet field encoding UMH identifier.

Figure 6: PIM IPv6 TreeInfo item

8.3. Incremental deployment

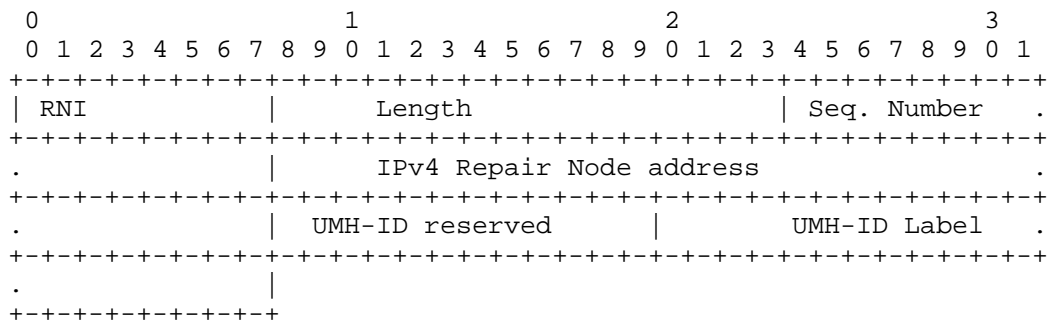
Joins with a RNI can be forwarded through legacy nodes if the Transitive Attribute (see [RFC5384]) has the F bit set to 1. It is up to the network operator to determine this. The DTN functionality can be deployed incrementally as long as the node detecting the failure and Repair Nodes support it.

9. mLDP Specific TN Components

In this section we are documenting the mLDP specific data-structures and procedures (if they are different from the generic procedures are defined in this document). As described in this document, TN packets are UDP/IP packets sent via unicast to its destination. The UDP port number for mLDP is set to the (to be) assigned IANA port number for mLDP-TN.

9.1. RNI item in mLDP Label Mapping

The RNI item for mLDP is encoded in a LDP MP Status TLV as documented in [RFC6388] section 5. A new LDP MP Status Value Element is created for this purpose and called the RNI Status. The RNI Status includes the router ID, sequence number and UMH Identifier. The UMH-ID can be locally unique identifier since its has only local significance on the Repair node. For mLDP the value that MUST be used is the Local Label associated with the UMH the mLDP Label Mapping is sent to. The RNI status is carried in Label Mapping messages and has the following format.



RNI Type: This 1 octet field assigned by IANA for RNI Status Value Element Types.

Length: This is a 2 octet field, describing the length of the Value, Length = 10 octets.

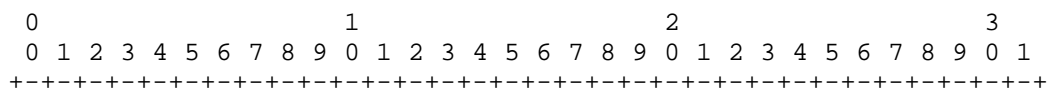
Sequence number: 2 octets long field, describing the sequence number of the sending Repair Node.

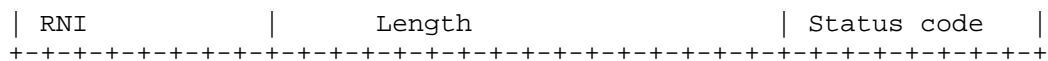
IPv4 Repair Node address: The IPv4 address of the Repair Node.

UMH-ID reserved: 12 bit field, reserved.

UMH-ID Label: This is a 20 bit field encoding a Label as UMH identifier.

Figure 7: mLDP RNI Status Value Element





MBB Type: Type 1 (to be assigned by IANA)

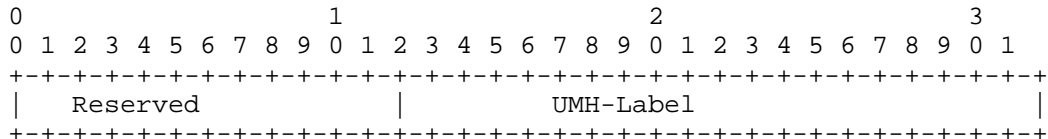
Length: 1

Status code: 1 = MBB request

2 = MBB ack

9.2. Tree Information Item

A TN packet contains one or more TreeInfo items that allows a Merge Node to identify which tree(s) and interface(s) are effected by the TN. The same encoding is used for DTN and UTN packets. Following [RFC6388], mLDP will assign a unique Label to each upstream node per MP-LSP. This label identifies the UMH AND the LSP. Since we are using a label to identify the UMH and LSP, there is no need to define a IPv4 and IPv6 specific encoding. The Label included in the DTN MUST be taken from the RNI that was signalled for that tree. The Label for UTN packets is the Local Label that was allocated for that tree. The TreeInfo item has the following format:



Reserved: This is a 12 bits field, set to zero on sending, ignored when received.

UMH-Label: This is a 20 bit field encoding MPLS Label of the UMH.

Figure 8: mLDP TreeInfo item

10. Acknowledgements

The authors would like to thank Stefan Olofsson, Javed Asghar and Greg Sheperd for their comments on the draft.

11. IANA Considerations

IANA is requested to allocate UDP port numbers to TN messages. One port number for TN in IP/PIM context, and another one for MPLS/mLDP context. The separation of UDP port numbers between IP and MPLS is requested to prevent problems when a PIM multicast tree is transported partly through an mLDP multicast tree.

IANA is requested to allocate a value from "PIM Join Attribute" to make routers capable to advertisement their Tree Notification capability.

IANA is requested to allocate a value from "PIM Join Attribute Types" for TN's join command extra information.

A new IANA registry is needed for "TN option TLVs". This describes the types of TLVs containing extra options for TN messages.

12. Security Considerations

Two types of security problems can be foreseen by the authors:

- o Handling illegally injected TN packets
- o Handling replay attacks (re-injecting previous TN messages)
- o TN messages propagating outside an operator's domain

Illegal TN packets can be detected with authentication check. Providing authentication for TN messages is described in Section 6. Prevention of replay attacks needs authentication in combination with sequence numbering, which is also described at the same section.

Preventing TN messages that travel inline with data packets MUST be solved by nodes egressing the operator's domain. Solutions for IP and MPLS are described in sections Section 8 and Section 9, respectively.

13. References

13.1. Normative References

- [I-D.karan-mofrr]
Karan, A., Filsfils, C., Farinacci, D., Decraene, B.,
Leymann, N., and W. Henderickx, "Multicast only Fast Re-
Route", draft-karan-mofrr-02 (work in progress),

March 2012.

[RFC5384] Boers, A., Wijnands, I., and E. Rosen, "The Protocol Independent Multicast (PIM) Join Attribute Format", RFC 5384, November 2008.

[RFC6388] Wijnands, IJ., Minei, I., Kompella, K., and B. Thomas, "Label Distribution Protocol Extensions for Point-to-Multipoint and Multipoint-to-Multipoint Label Switched Paths", RFC 6388, November 2011.

13.2. Informative References

[I-D.atlas-rtgwg-mrt-mc-arch]
Atlas, A., Kebler, R., Wijnands, I., Csaszar, A., and G. Envedi, "An Architecture for Multicast Protection Using Maximally Redundant Trees", draft-atlas-rtgwg-mrt-mc-arch-02 (work in progress), July 2013.

Authors' Addresses

IJsbrand Wijnands (editor)
Cisco
De kleetlaan 6a
Diegem, 1831
Belgium

Phone:
Email: ice@cisco.com

Luc De Ghein
Cisco
De kleetlaan 6a
Diegem, 1831
Belgium

Phone:
Email: ldeghein@cisco.com

Gabor Sandor Enyedi (editor)
Ericsson
Konyves Kalman Krt 11/B
Budapest, 1097
Hungary

Phone:
Email: Gabor.Sandor.Enyedi@ericsson.com

Andras Csaszar
Ericsson
Konyves Kalman Krt 11/B
Budapest, 1097
Hungary

Phone:
Email: Andras.Csaszar@ericsson.com

Jeff Tantsura
Ericsson
300 Holger Way
San Jose, California 95134
USA

Email: Jeff.Tantsura@ericsson.com

